UNIVERSIDAD DE CARABOBO
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA DE
TELECOMUNICACIONES
DEPARTAMENTO DE SEÑALES Y SISTEMAS

DESARROLLO DE SOFTWARE LIBRE INTERACTIVO PARA REALIZAR ANÁLISIS ESPECTRAL DE VOZ

YANINA A. PERDOMO V.

Bárbula, 17 de Julio del 2015

UNIVERSIDAD DE CARABOBO
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA DE
TELECOMUNICACIONES
DEPARTAMENTO DE SEÑALES Y SISTEMAS

DESARROLLO DE SOFTWARE LIBRE INTERACTIVO PARA REALIZAR ANÁLISIS ESPECTRAL DE VOZ

TRABAJO ESPECIAL DE GRADO PRESENTADO ANTE LA ILUSTRE UNIVERSIDAD DE CARABOBO PARA OPTAR AL TÍTULO DE INGENIERO DE TELECOMUNICACIONES

YANINA A. PERDOMO V.

Bárbula, 17 de Julio del 2015

UNIVERSIDAD DE CARABOBO
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA DE TELECOMUNICACIONES
DEPARTAMENTO DE SEÑALES Y SISTEMAS

CERTIFICADO DE APROBACIÓN

Los abajo firmantes miembros del jurado asignado para evaluar el trabajo especial de grado titulado «DESARROLLO DE SOFTWARE LIBRE INTERACTIVO PARA REALIZAR ANÁLISIS ESPECTRAL DE VOZ», realizado por el bachiller YANINA A. PERDOMO V., cédula de identidad 19.580.741, hemos decidido otorgar la máxima calificación y la mención honorífica al presente trabajo, con base a los siguientes motivos:

1) La estudiante mostró dominio en los contenidos teóricos desarrollados en la investigación a través de una impecable presentación. 2) La estudiante desarrolló un alcance que sobrepasó los límites iniciales planteados, estructurando una herramienta muy útil para futuras investigaciones en el área de análisis espectral de señales de voz.

Firma
Prof. AHMAD OSMAN
TUTOR

Firma
Prof. PAULINO DEL PINO
JURADO

Firma
Prof. ELIMAR HERNÁNDEZ
JURADO

Bárbula, 17 de Julio del 2015

Dedicatoria

A Dios, porque es Él quien produce en nosotros
tanto el querer como el hacer,
según su buena voluntad

YANINA A. PERDOMO V.

Agradecimientos

A Dios, por ser mi fuente de sabiduría, conocimiento y fortaleza.

A mis padres, por creer siempre en mi capacidad para lograr mis metas, de manera muy especial a mi madre por ser mi buen ejemplo a seguir, mi compañera de lucha en todo momento e impulsarme a pensar siempre en grande.

A mis hermanos, porque se han convertido en unos segundos padres para mí, brindándome todo el amor y el apoyo para seguir adelante, en especial mi hermana y mejor amiga que siempre ha tenido un corazón paciente para escucharme y brindarme los mejores consejos.

A mis sobrinos: Manuel, Camila, Sebastián y Victoria por ser mi fuente principal de alegría.

A la Ilustre Universidad de Carabobo, por brindarme las facilidades para obtener mi formación académica.

A Ahmad Osman, tutor acádemico, por todo el apoyo y asesoramiento durante el desarrollo de este Trabajo Especial de Grado.

A los profesores de la escuela de Telecomunicaciones, por toda la formación recibida, en especial al profesor Carlos Mejías por los acertados consejos durante el desarrollo de esta investigación.

A Jesús Montes, por su apoyo, comprensión y palabras de aliento durante este tiempo.

A mis compañeros, especialmente Ronald, Rossana, Jennifer, Manuel, Daniel y Yutzani, por todas las experiencias durante nuestra formación, dignas de recordar con una sonrisa y que hicieron de este, un tiempo más agradable.

A quien además de ser mi compañera de estudios, hoy en día es mi amiga: Yutzani, por todo el apoyo incondicional e invaluable durante el desarrollo de este trabajo.

Índice general

Índice de figuras

Indice de tablas

Acrónimos

DSP	**D**igital **S**ignal **P**rocessing
TDF	**T**ransformada **D**iscreta de **F**ourier
FFT	**F**ast **F**ourier **T**ransform
DIT	**D**ecimation **I**n **T**ime
DIF	**D**ecimation **I**n **F**requency
LPC	**L**inear **P**redictive **C**oding
STFT	**S**hort **T**ime **F**ourier **T**ransform
PFA	**P**rime **F**actor **A**lgorithm

DESARROLLO DE SOFTWARE LIBRE INTERACTIVO PARA REALIZAR ANÁLISIS ESPECTRAL DE VOZ

por

YANINA A. PERDOMO V.

Presentado en el Departamento de Señales y Sistemas
de la Escuela de Ingeniería en Telecomunicaciones
el 17 de Julio del 2015 para optar al Título de
Ingeniero de Telecomunicaciones

RESUMEN

En la presente investigación se planteó la creación un software libre interactivo para caracterizar el espectro de las señales de voz usando algoritmos eficientes para el cómputo de la TDF. Para esto se realizó la revisión del estado del arte de los algoritmos FFT, se seleccionaron los más significativos, se implementaron en Python mediante módulos importables y se diseñó una interfaz de usuario para el uso interactivo de los algoritmos. Fue creada una librería con cuatro algoritmos de FFT, en forma de módulos importables a Python. Se generó una herramienta interactiva, bajo lenguaje Python, para análisis espectral de voz que permite la determinación de espectro de potencia, estimación de formantes, estimación de pitch y extracción de envolvente espectral. Se realizaron pruebas de estimación de parámetros de voz, obteniendo resultados relevantes en la detección del pitch.

Palabras Claves: Análisis espectral, señal de voz, Transformada Discreta de Fourier

Tutor: AHMAD OSMAN
Profesor del Departamento de Señales y Sistemas
Escuela de Telecomunicaciones. Facultad de Ingeniería

Capítulo I

Introducción

1.1. Motivación

La voz humana ha pasado a ser un importante objeto de investigación en distintas especialidades, ya que se ha demostrado que mediante el análisis de la señal de voz producida por las personas se pueden determinar características como el sexo, la edad e incluso, investigaciones recientes en el área de la ingeniería biomédica demuestran que es posible identificar patologías en las personas y enfermedades neurológicas. Lo anterior constituye un método alternativo, no invasivo y además menos costoso, que permite la detección temprana e incluso la evaluación a distancia del progreso de determinadas patologías.[6],[7], [8]

El diagnóstico está basado en la extracción de las características de las señales de voz que pueden determinar la calidad de la misma, esto se realiza a través de técnicas de procesamiento digital de señales; habitualmente los parámetros de interés son el espectro de potencia, la frecuencia fundamental y los formantes, que son perceptibles en el dominio de la frecuencia y a partir de los cuales pueden ser determinadas otras características más específicas e incluso las perturbaciones de la voz. [9]

Existen diferentes herramientas para realizar el análisis espectral de las señales de voz, una de ellas es la Transformada Discreta Fourier, la cual es útil para el

análisis de señales digitales; y cuyo rendimiento ha sido probado en distintas investigaciones que han logrado la extracción de parámetros mediante el estudio del espectro en frecuencia obtenido a través de su aplicación a señales de voz bajo distintas condiciones de análisis. Ahora bien, las señales de voz son no estacionarias y su análisis se realiza en intervalos cortos de la señal en los que se puede asumir que la señal es estacionaria; la técnica usada para este análisis se conoce como Transformada de Fourier de Tiempo Reducido (STFT). [9] [10] [11] [12]

Aunado a esto la TDF puede ser determinada a través de algoritmos más eficientes, que se conocen de manera colectiva como Transformada Rápida de Fourier (FFT), cuya ventaja principal es la eficiencia derivada de la reducción del número de operaciones a realizar y por ende, el tiempo de cómputo de los resultados. [13], [14], [15].

Ahora bien, los parámetros espectrales de las señales de voz, se pueden estimar mediante técnicas basadas en la TDF. Es por ello que se considera conveniente desarrollar un software libre interactivo para realizar análisis espectral de voz, la cual es la propuesta de la presente investigación

Con el desarrollo de la investigación se realiza un aporte al Departamento de Señales y Sistemas de la Escuela de Telecomunicaciones de la Facultad de Ingeniería de la Universidad de Carabobo, pues el desarrollo del software planteado constituirá una herramienta de apoyo para la realización de las prácticas del Laboratorio de Procesamiento Digital de Señales; además puede ser el punto de partida para investigaciones o desarrollo de aplicaciones más específicas en el área de Procesamiento Digital de Señales de Voz.

1.2. Objetivos

1.2.1. Objetivo General

Desarrollar un software libre interactivo para caracterizar el espectro de las señales de voz usando algoritmos que permitan aproximar la Transformada Discreta

de Fourier(TDF), para el estudio de la voz humana.

1.2.2. Objetivos Específicos

1. Revisar el estado del arte de los algoritmos que permitan aproximar la Transformada Discreta de Fourier(TDF).
2. Seleccionar los algoritmos más significativos que permitan aproximar la Transformada Discreta de Fourier (TDF).
3. Implementar los algoritmos seleccionados para la realización del análisis espectral de las señales de voz.
4. Diseñar una interfaz con el usuario, para el manejo interactivo de cada uno de los algoritmos codificados.

1.3. Alcances

Se produjo una herramienta interactiva que incluye cuatro algoritmos para determinar la Transformada Discreta de Fourier (TDF), 16 funciones ventana, cuya longitud y solapamiento son datos de entrada, cálculo de espectro de potencia, cálculo del espectrograma y estimación de pitch y formantes.

Capítulo II

Marco conceptual

2.1. Características de las Señales de Voz

De acuerdo con la acción de las cuerdas vocales, las señales de voz pueden ser clasificadas en vocales y no vocales. [16]

En las señales vocales, que consisten en la generación de sonidos sonoros, el tracto vocal tiene un comportamiento similar al de una cavidad resonante, el sonido se produce como efecto de la vibración de las cuerdas vocales, las cuales modifican el área de la traquea al abrirse y cerrarse produciendo un tren de pulsos casi periódicos. En el dominio de la frecuencia estas señales están conformadas por armónicos, como consecuencia de su casi periodicidad en el dominio del tiempo, y una envolvente espectral debida al tracto vocal. [16], [17].

En las señales no vocales, que consisten en la generación de sonidos sordos, las cuerdas vocales permanecen abiertas y el aire fluye libremente por el tracto vocal, por lo que están formadas por una contribución desordenada de componentes frecuenciales y presentan una aleatoriedad similar a la del ruido blanco. [16], [17].

2.1.1. Frecuencia fundamental y pitch

La frecuencia fundamental es la frecuencia de vibración de las cuerdas vocales y se puede interpretar como el número de veces que estas se abren y cierran por segundo. También es conocida también como tono habitual, pues es el nivel óptimo en el cual la voz es producida sin esfuerzo y sin tensión en la laringe. [18],[19], [20].

Esta frecuencia varía de acuerdo al sujeto y las características de longitud, grosor y tensión de sus cuerdas vocales, sin embargo los valores típicos en adultos se encuentran entre 137Hz para los hombres y 207Hz para las mujeres. Su determinación resulta de interés para identificar los patrones de vibración de las cuerdas vocales, pudiendo detectar alguna alteración de los mismos en el caso de que existan patologías. [21], [18].

La frecuencia fundamental se corresponde con el tono percibido o pitch, el cual es la percepción del oyente a cambios de frecuencia, es un fenómeno psicológico, mientras que la frecuencia fundamental es un hecho de la física y se ve afectado por las características del tracto vocal del sujeto, tal como se explicó anteriormente. Sin embargo, en el caso de tratamiento de las señales de voz, la frecuencia fundamental y el pitch son considerados como un solo concepto. [22].

2.1.2. Formantes

Los formantes del habla se corresponden con las resonancias de baja frecuencia en el tracto vocal, son los máximos que se producen en la envolvente del espectro de potencia, por lo que pueden ser identificados en un espectrograma; esta característica solo se observa en las señales vocales, pues la no vocales tienen una estructura ruidosa como se describió anteriormente. Los tres primeros formantes de la señal de voz (denotados como $\mathrm{F1}$, $\mathrm{F2}$ y $\mathrm{F3}$) contienen suficiente información acerca de la señal de voz y han sido considerados como la fuente principal de información espectral. [17] [23] [5]

2.2. Transformada Discreta de Fourier (TDF)

En tiempo discreto el par transformado de Fourier viene dado por las siguientes expresiones matemáticas:

$$x[n] = \frac{1}{2\pi}\int_{2\pi} X(e^{j\omega})e^{j\omega n}\, d\omega \tag{2.1}$$

$$X(e^{j\omega}) = \sum_{n=-\infty}^{+\infty} x[n]\, e^{-j\omega n}, \tag{2.2}$$

definido para una secuencia no periódica de duración finita $x[n]$. La ecuación 2.1 se conoce como ecuación de síntesis, mediante ésta es posible reconstruir la señal como una combinación de exponenciales complejas; mientras que la ecuación 2.2 representa la ecuación de análisis, la cual proporciona la información de cómo la señal $x[n]$ se compone de exponenciales complejas de diferentes frecuencias.

Ahora bien, $X(e^{j\omega})$ es una función contínua y periódica en el dominio de la frecuencia, la definición de la Transformada Discreta de Fourier surge al muestrear esta función en intervalos iguales de frecuencia para obtener N muestras equiespaciadas de la misma. La expresión de la Transformada Discreta de Fourier se muestra a continuación:

$$X[k] = \sum_{n=0}^{N-1} x[n]\, W_N^{kn}, \tag{2.3}$$

donde

$$W_N = e^{-j\frac{2\pi}{N}} \tag{2.4}$$

El término W_N^{kn}, el cual es la raíz $N-ésima$ de la unidad, es una función periódica de kn con periodo N, por lo que existirán N raíces sobre la circunferencia unitaria del plano complejo, que también serán periódicas. Cada una de estas raíces recibe el nombre de factor de rotación debido a que la multiplicación de un número por una de ellas cambia la fase de ese número sin cambiar su magnitud.

2.3. Cómputo de la Transformada Discreta de Fourier

El cálculo de la TDF de una secuencia de entrada $x[n]$ de N muestras, por el método directo requiere N^2 multiplicaciones complejas y $N(N-1)$ sumas complejas, si la función es expresada en función de operaciones con números reales, requiere $4N^2$ multiplicaciones reales y $N(4N-2)$ sumas reales. [24], [25], [3].

La TDF tiene una enorme capacidad para mejorar su eficiencia aritmética, debido a la periodicidad, simetría y ortogonalidad de las funciones base y la relación con la convolución, por esta razón la aplicación del método directo para determinar la TDF es considerado básicamente ineficiente porque no explota las propiedades de simetría y periodicidad del factor de fase W_N.[26],[2].

El desarrollo de algoritmos rápidos usualmente consiste en usar las propiedades especiales del algoritmo de interés para eliminar las operaciones redundantes o innecesarias de la implementación directa. [2].

La potencia real del método FFT es que, a menudo, la división se puede aplicar de forma recursiva a los subproblemas, lo que conduce a una reducción del orden de complejidad.[27].

2.3.1. Algoritmos para N potencia entera de 2

Denominados también FFT de base 2^ν, aplicables cuando el número de muestras de la secuencia de entrada es una potencia entera de 2, de allí surgen algoritmos de base $2, 4, 8, 16, \cdots$, lo más importantes son los algoritmos FFT de base 2, algoritmos FFT de base 4 y el algoritmo de bases o raíces partidas, en esta sección se discutirán los algoritmos de base 2. [27].

2.3.1.1. Diezmado en Tiempo (DIT)

Los algoritmos de diezmado en el tiempo se basan en determinar $X[k]$, mediante la división de la secuencia de entrada $x[n]$ en subsecuencias diezmadas con diferente fase, para así determinar TDF más pequeñas.[27].

Para determinar la TDF se procede separando las muestras pares e impares de la secuencia de entrada $x[n]$ de N muestras, obteniendo dos secuencias de longitud $\frac{N}{2}$, de esta manera la TDF quedaría expresada como se muestra a continuación,

$$X[k] = \sum_{r=0}^{(N/2)-1} x[2r]W_{N/2}^{rk} + W_N^k \sum_{r=0}^{(N/2)-1} x[2r+1]W_{N/2}^{rk} \tag{2.5}$$

tomando $n = 2r$ para n par y $n = 2r + 1$ para n impar. Reescribiendo la ecuación (2.5), la Transformada Discreta de Fourier de la secuencia $x[n]$ puede expresarse como la suma de dos funciones, una correspondiente a la parte par y la otra correspondiente a la parte impar de la secuencia. Estas sumas son TDF de $(\frac{N}{2})$ puntos de $x[n]$ y se calculan para valores de $k = 0, 1, ...\frac{N}{2} - 1$ ya que son periódicas con periodo igual a $(\frac{N}{2})$. Finalmente la expresión de la secuencia transformada válida para valores de $k = 0, 1, ..., N - 1$, será:

$$X[k] = G[k] + W_N^k H[k], \tag{2.6}$$

El cálculo de la TDF mediante este algoritmo requiere un máximo de $N+2(N/2)^2$ operaciones. Ahora bien, por ser N potencia de 2, se puede dividir el cálculo de las dos TDF de $N/2$ puntos, aplicando nuevamente el algoritmo para cada una. La descomposición del cálculo se puede repetir hasta que solo sea necesario determinar transformadas de 2 puntos. Esta descomposición genera un total de $\nu = \log_2 N$ etapas de cálculo, por lo que el máximo de sumas y multiplicaciones a realizar es $N\log_2 N$. El grafo de flujo del cálculo de la TDF mediante el algoritmo descrito se muestra en la figura (2.1), para un valor de $N = 16$. [3],[1].

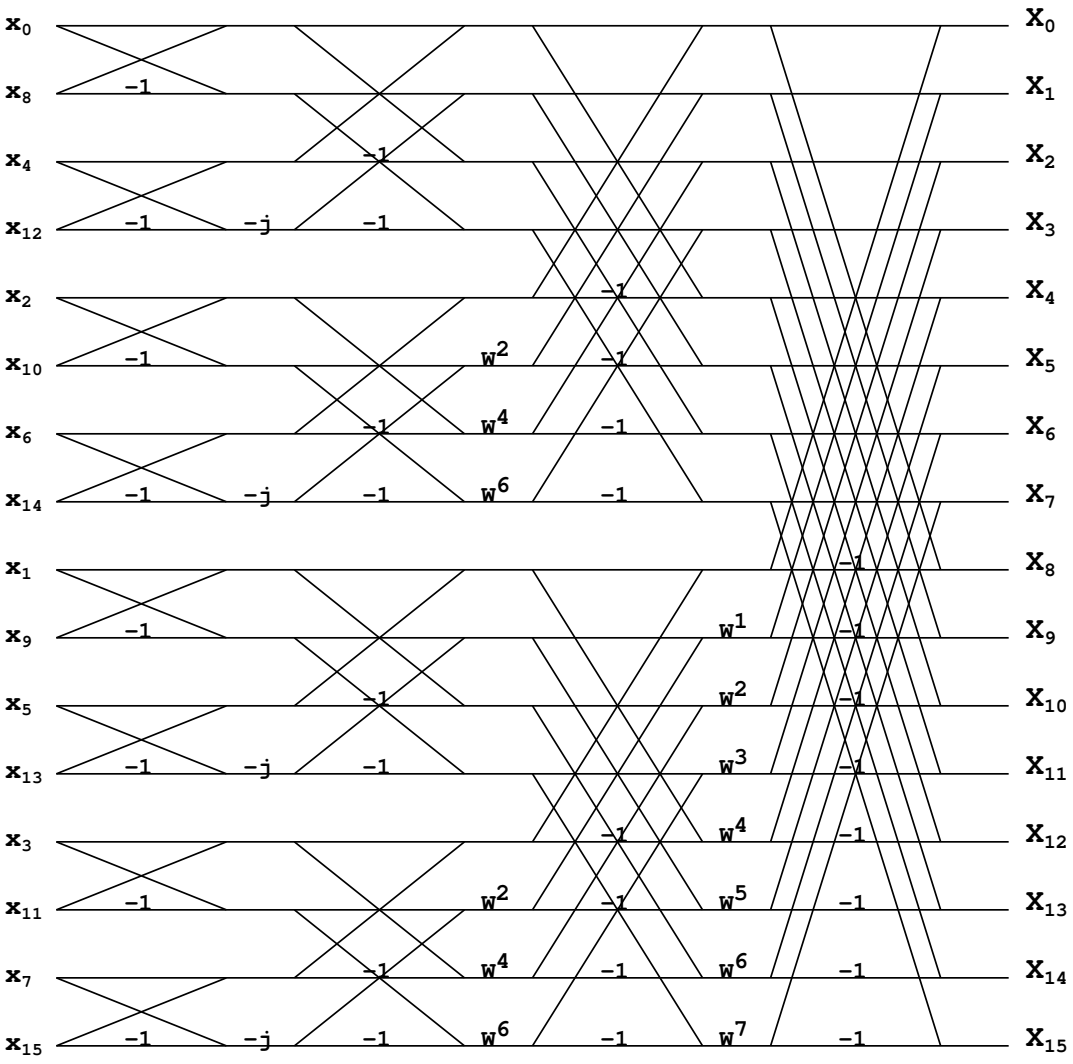

Figura 2.1: Grafo de flujo de la descomposición del cálculo de la TDF para N $= 16$, mediante DIT de base-2. Fuente: [1]

2.3.1.2. Diezmado en Frecuencia (DIF)

El algoritmo de diezmado en frecuencia se basa en realizar el cálculo de TDF, dividiendo la secuencia X[k] en subsecuencias más pequeñas; este algoritmo surge de aplicar un enfoque dual en la estructura del algoritmo presentado por Cooley y Tukey, el cual consiste en realizar diezmado en el tiempo. [3],[27].

Para una secuencia de entrada x[n] de longitud N $= 2^{\nu}$ la TFD se determina

dividiendo la secuencia de salida $X[k]$ en sus muestras pares e impares, con un cambio de variable $k = 2r$ y $k = 2r + 1$ en cada caso. Además, considerando la periodicidad del factor de fase, para valores de $r = 0, 1, ..., (N/2) - 1$, se obtiene

$$X[2r] = \sum_{n=0}^{(N/2)-1} (x[n] + x[n + (N/2)])W_{N/2}^{rn} \tag{2.7}$$

$$X[2r + 1] = \sum_{n=0}^{(N/2)-1} (x[n] - x[n + (N/2)])W_{N}^{n}W_{N/2}^{nr} \tag{2.8}$$

La ecuación (2.7) es la TDF de la secuencia resultante de la suma de la primera y segunda mitad de la secuencia de entrada, mientras que la ecuación (2.8) es la TDF de la secuencia que resulta de restar la segunda mitad de la secuencia de entrada de la primera mitad, y multiplicarla por el factor W_N^n. Ambas TDF son de $N/2$ puntos. [3].

Tomando $g[n] = x[n] + x[n + N/2]$ y $h[n] = x[n] - x[n + N/2]$, la TDF se puede determinar formando ambas secuencias $g[n]$ y $h[n]$, multiplicando luego por el factor de rotación a la secuencia que corresponde y finalmente tomando la TDF de $N/2$ puntos. [3]

Ahora bien, al igual que en el caso de diezmado en el tiempo de base 2, la división de la secuencia se puede repetir $\nu = \log_2 N$ veces, con lo cual el cálculo se reduce a determinar transformadas de 2 puntos. El número total de operaciones serán $(N/2)\text{Log}_2 N$ multiplicaciones complejas y $N\text{Log}_2 N$ sumas complejas. En la figura (2.2) se muestra el grafo de flujo que representa el cálculo de la TDF de una secuencia de $N = 16$ mediante el algoritmo descrito. [3], [1].

En el apéndice A de este documento se encuentran disponibles dos códigos desarrollados bajo lenguaje Fortran, que permiten la determinación de la TDF, un código mediante DIF y otro mediante DIT, con la única restricción que la longitud de la secuencia de entrada sea potencia entera de dos, los datos de entrada son la señal a transformar en dos arreglos: parte imaginaria y parte real, N y ν. Los datos de salida son guardados en los arreglos correspondientes a la entrada, lo cual refleja

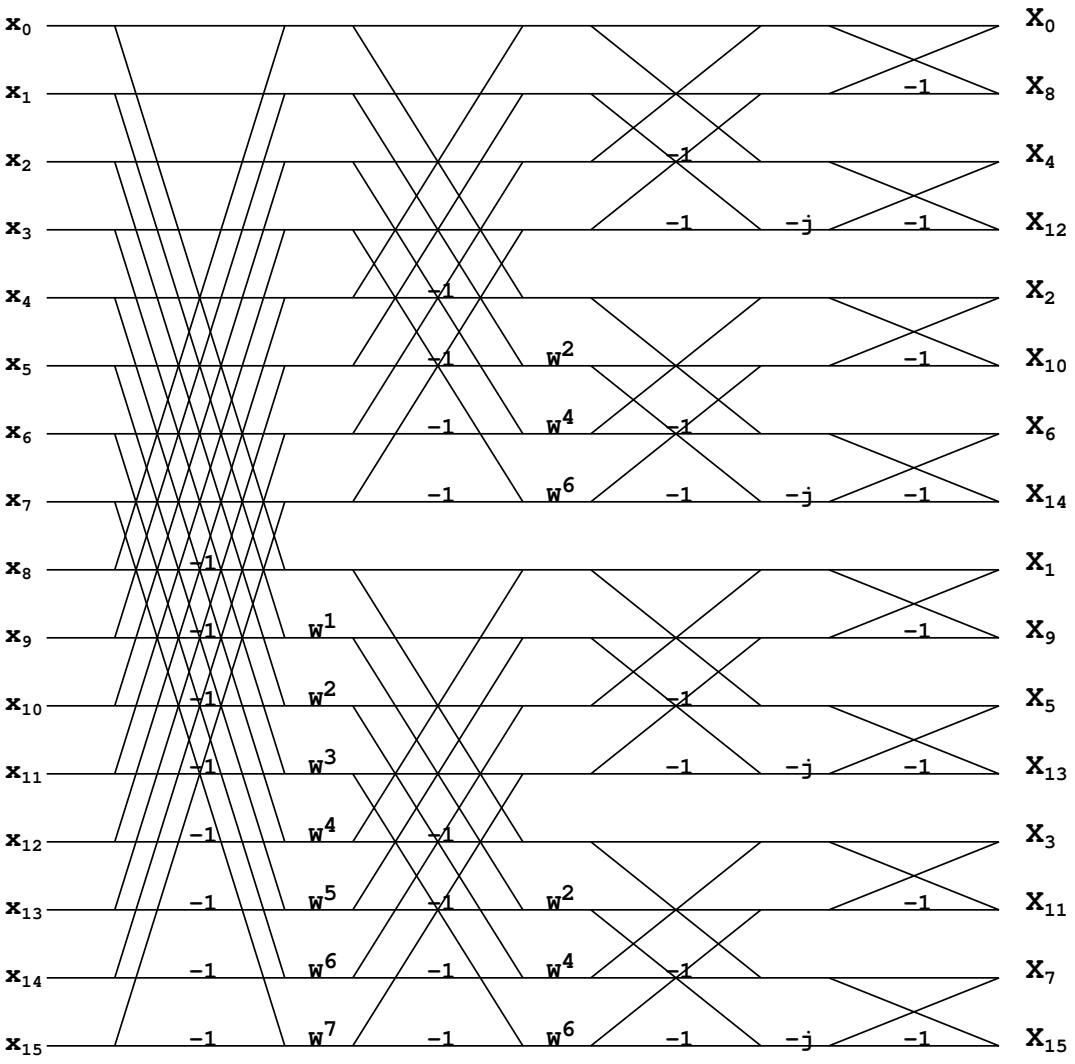

Figura 2.2: Grafo de flujo de la descomposición del cálculo de la TDF para $N = 16$, mediante DIF de base 2. Fuente [1]

un esquema de cómputo en el mismo lugar, el cual es un procedimiento útil para almacenar los datos. [3]

2.3.2. Algoritmo de Factores Primos (PFA)

En este algoritmo se hace uso del mapeo de Good para convertir la TDF unidimensional de longitud $N = N_1N_2$ en una TDF bidimensional de tamaño $N =$

N_1xN_2, y luego se calcula esta TDF 2D por filas y por columnas, usando los algoritmos más eficientes en cada dimensión. [27].

El mapeo de la TDF se realiza cambiando los índices n y k de acuerdo a las siguientes ecuaciones:

$$n = ((N_2 n_1 + N_1 n_2))_N \tag{2.9}$$

$$k = ((pN_2 k_1 + qN_1 k_2))_N \tag{2.10}$$

donde p y q son las soluciones de las ecuaciones

$$((pN_2 = 1))_{N_1}, \quad (p < N_1) \tag{2.11}$$

$$((qN_1 = 1))_{N_2}, \quad (q < N_2) \tag{2.12}$$

Al usar este esquema de indexación, se obtiene la nueva expresión para la TDF será

$$X[k_1, k_2] = \sum_{n_2=0}^{N_2-1} \sum_{n_1=0}^{N_1-1} x[n_1, n_2] W_{N_1}^{n_1 k_1} W_{N_2}^{n_2 k_2} \tag{2.13}$$

la cual es una TDF bidimensional sin factores de rotación y las sumatorias pueden ser determinadas en cualquier orden.Este algoritmo se puede aplicar de manera recursiva para más de dos factores,con la única limitación de que deben ser primos entre sí. [2] ,[27].

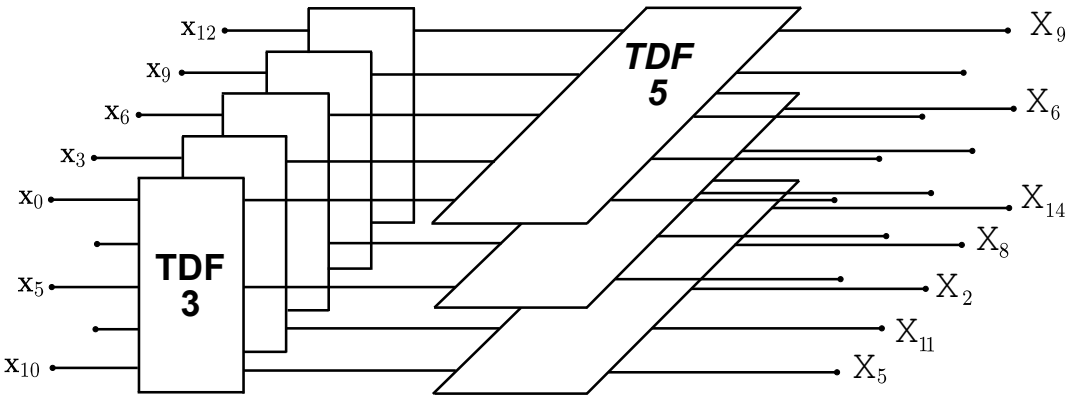

Figura 2.3: Esquema de PFA para $N = 15$. Fuente [2]

En la figura (2.3) se muestra la vista esquemática de la TDF de una secuencia de entrada de $N = 3x5$, se observa que el procedimiento es: una vez hecho el mapeo con las ecuaciones (2.9) y (2.10), se realiza la transformada resolviendo primero por columnas cinco TDF de tres puntos cada una, y luego por filas, tres TDF de cinco puntos. Sin factores de rotación entre etapas. Claramente, también puede resolverse primero por filas y luego por columnas. [27].

En el A de este documento se encuentra un código de aplicación para el PFA, desarrollado bajo lenguaje Fortran y con un esquema de cómputo en el mismo lugar, cuya estructura consiste en realizar el mapeo y luego determinar la TDF para cada factor primo; para esto cuenta con módulos para transformar secuencias de longitud 2,3,4,5,6,7,8,9, y 16. Los datos de entrada además de la secuencia a transformar y el número de muestras, son el números de factores primos en el que se descompone N y un arreglo con los factores primos. La restricción para la aplicación de este algoritmo es que los factores deben ser primos entre sí.

2.3.3. Algoritmo de Goertzel

El algoritmo se basa en el uso de la periodicidad de la secuencia W_N^{kn} para reducir los cálculos en la determinación de la TDF de la secuencia de entrada $x[n]$ de N muestras.[3].

Se define la secuencia

$$y_k[n] = \sum_{r=-\infty}^{\infty} x[r] W_N^{-k(n-r)} u[n-r], \tag{2.14}$$

tomando en cuenta que $x[n] = 0$ para $n < 0$ y $n >= N$, la TDF se obtiene evaluando la ecuación (2.14) en $n = N$, por lo tanto

$$X[k] = y_k[n]|_{n=N} \tag{2.15}$$

La ecuación 2.14 se puede interpretar como la convolución discreta de la secuencia de duración finita $x[n]$ con la secuencia $W_N^{-kn} u[n]$. Por lo que $X[k]$ es la

salida de un sistema cuya respuesta al impulso es $W_N^{-kn}u[n]$, cuando $n = N$. Usando esta definición, denominada algoritmo de Goertzel de primer orden, el cálculo requiere $4N$ multiplicaciones reales y $4N$ sumas reales, para cada valor de k, lo cual es menos eficiente que el método directo, sin embargo como los coeficientes W_N^{-kn} son determinados mediante recursión, se evita el cálculo y almacenamiento de esas cantidades. [3].

La función de transferencia del sistema mencionado es

$$H_k(z) = \frac{1}{1 - W_N^{-k} z^{-1}} \tag{2.16}$$

La función (2.16) es modificada, multiplicando el numerador y denominador por el factor $(1 - W_N^k z^{-1})$ haciéndola de segundo orden,con el objetivo de eliminar las multiplicaciones complejas y así reducir el número de multiplicaciones por un factor de 2. La nueva función de transferencia será

$$H_k(z) = \frac{1 - W_N^k z^{-1}}{1 - 2\cos(2\pi k/N) z^{-1} + z^{-2}} \tag{2.17}$$

El grafo de flujo correspondiente a la función de transferencia de la ecuación (2.17), se muestra en la figura (2.4). El número de operaciones requeridas para la determinación de la TDF de N muestras con este algoritmo de Goertzel de segundo orden, es de $2(N + 2)$ multiplicaciones reales y $4(N + 1)$ sumas reales, lo cual representa aproximadamente la mitad del número de multiplicaciones requeridas en el método directo. Este método resulta útil cuando se necesita determinar solo algunos valores de $X[k]$. Tanto el método directo como el algoritmo de Goertzel son más eficientes que los algoritmos de diezmado cuando el número de muestras a determinar es menor que $\log_2 N$, siendo N potencia de 2.

En el apéndice A se encuentra un código en lenguaje Fortran para aplicar este algoritmo a una secuencia, los datos de entrada son la secuencia y su longitud, con respecto a esta última no existen restricciones y, en general, este algoritmo es aplicable para cualquier conjunto de muestras.

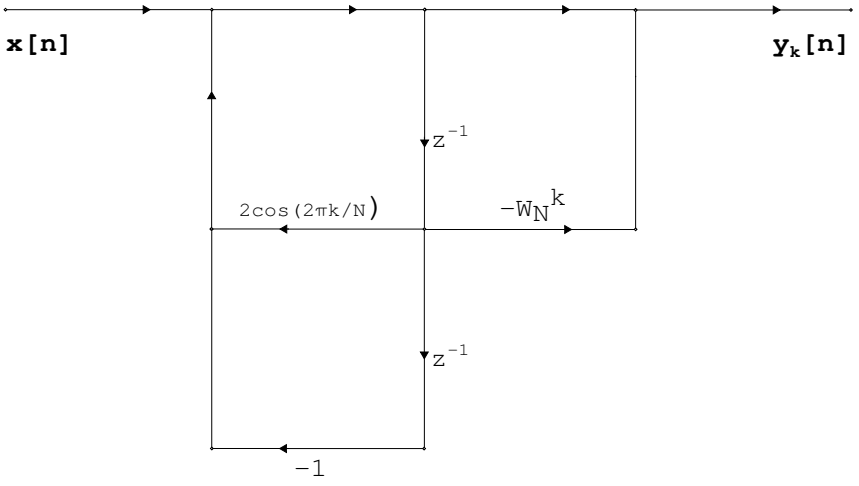

Figura 2.4: Grafo de flujo para el cálculo recursivo de segundo orden de X[k]. Fuente [3]

2.4. Transformada de Fourier de Tiempo Reducido (STFT)

La STFT es una poderosa herramienta de propósito general para procesamiento de señales de audio. Esto define una clase particularmente útil de las distribuciones tiempo-frecuencia que especifican la amplitud compleja en función del tiempo y la frecuencia para cualquier señal. [23]

El principio de esta herramienta, es aplicar una función ventana a la señal de entrada $x[n]$ y determinar la transformada de Fourier de cada una de las porciones obtenidas.De esta manera, la STFT

$$X[n,\lambda) = \sum_{m=-\infty}^{\infty} x[n+m]w[m]e^{-j\lambda m} \tag{2.18}$$

donde $w[m]$ es una función ventana, la cual tiene un origen estacionario y a medida que n cambia, la señal de desliza por la ventana de forma que en cada valor de n se ve una parte diferente de la señal. [3]

La secuencia unidimensional $x[n]$, se convierte en una función bidimensional de la variable temporal discreta n y de la variable de frecuencia λ que es contínua. [3]

La ecuación 2.18 es la transformada de Fourier en tiempo discreto de $x[n+m]w[m]$, muestreando 2.18 en N frecuencias equiespaciadas,tomando $\lambda_k = 2\pi k/N$ se obtiene:

$$X[n,k] = \sum_{m=0}^{L-1} x[n+m]w[m]e^{-j(\frac{2\pi}{N})km}, \quad 0 \leqslant k \leqslant N-1 \tag{2.19}$$

donde L es la longitud de la función ventana y $N \leqslant L$. De esta manera, 2.19 es la Transformada Discreta Dependiente del Tiempo; y puede ser interpretada también como el resultado de pasar la TDF de la señal a través de un filtro cuya respuesta en frecuencia es la TDF de la función ventana.

Si, además, 2.19 se muestrea en el tiempo para la región de soporte $0 \leqslant k \leqslant L-1$ de la ventana $w[m]$, se puede definir como

$$X_r[k] = X[rR,k] = \sum_{m=0}^{L-1} x[rR+m]w[m]e^{-j(\frac{2\pi}{N})km} \tag{2.20}$$

donde r y k son enteros tales que $-\infty < r < \infty$ y $0 \leqslant k \leqslant N-1$, r es el índice de la trama y R es el tamaño del salto de la posición de la ventana $w[m]$.

De acuerdo con la definición presentada, 2.20 es simplemente una secuencia de TDF de N puntos de segmentos de datos enventanados, por lo que el espectro resultante para cada porción tendrá N frecuencias equiespaciadas. [3],[28].

En el análisis de señales de voz el ajuste de los parámetros de STFT se hace pensando en la medición de las características de la voz en un intervalo de tiempo reducido, procurando lograr el equilibrio entre la resolución de los armónicos y la detección tanto del pitch como de las variaciones de los formantes.[23]

2.5. Enventanado de Señales para Análisis Espectral

En el análisis espectral de señales ocurridas naturalmente, casi siempre se analiza un segmento corto de señal (10 a $40[ms]$), en lugar de toda la señal. Por lo tanto, el primer paso en el análisis tiempo-frecuencia de una señal de audio consiste en

la segmentación de la misma; la forma correcta de extraer los segmentos cortos es multiplicar la señal por una función ventana.

Una función ventana no es más que una envolvente específica que se aplica a la señal a analizar. En general, la mayoría de las ventanas usadas en análisis espectral, tienen características pasa bajos en el dominio de la frecuencia y una forma muy similar a la curva de Gauss en el tiempo. El objetivo principal de usar una ventana de desvanecimiento es evitar las discontinuidades abruptas en los bordes durante la segmentación de la señal.

Las dos características más importantes relacionadas a una ventana de una longitud dada son: el ancho de su banda de paso (lóbulo principal) y la atenuación en su banda de rechazo (lóbulos secundarios). El ancho del lóbulo principal impone un límite de la mínima distancia entre dos picos que deben ser resueltos en frecuencia, ya que si estos están más cerca que el ancho del lóbulo principal, serán integrados en un solo pico. Por supuesto, incrementando la longitud de la ventana se producen lóbulos principales estrechos, lo que ayuda con la resolución de picos espectrales muy cercanos.

Para una ventana de longitud M muestras, el tipo de ventana controla la supresión del lóbulo lateral (a expensas de la resolución cuando M es fijo) y la longitud controla la resolución en frecuencia. El beneficio principal de la elección de una buena función ventana de análisis de Fourier es la minimización de los lóbulos laterales que causan diafonía en el espectro estimado de una frecuencia a otra. [29] [23]

Cabe destacar que la resolución del espectro obtenido a partir de una señal enventanada, se ve limitada por el Principio de Incertidumbre de Heisenberg, de manera que no es posible lograr alta resolución en ambos dominios (temporal y frecuencial) de manera simultánea; por lo tanto la fijación de los parámetros de la función ventana dependerá del dominio que interese estudiar, ya que una ventana estrecha tendrá una alta resolución temporal, mientras que la resolución frecuencial será muy baja; y el empleo de una ventana ancha, tendrá una baja resolución temporal, mientras que la resolución frecuencial será alta. [29]

De acuerdo con la literatura consultada, las ventanas mayormente empleadas son: la ventana rectangular (cuya característica principal es que su amplitud es constante) la ventana de Hanning (atenúa la señal en los bordes), Hamming (similar a la ventana de Hanning) y Blackman; comparadas entre sí, la ventana rectangular y de Hanning tienen un lóbulo principal muy definido y baja atenuación de frecuencias parásitas, mientras que la ventana de Hanning y Blackman presentan características similares entre sí y una mayor atenuación de frecuencias parásitas que las mencionadas anteriormente. [30].

2.6. Técnicas de Extracción de Parámetros de Señales de Voz

2.6.1. Modelo de Producción de Voz

En la figura (2.5) se muestra el modelo (simplificado) del sistema de producción del habla, en este modelo la señal de voz ($x(n)$) es considerada como la salida de un sistema lineal, que es excitado por un tren de impulsos en el caso de voz sonora o ruido en el caso de voz sorda. Los parámetros del modelo son la selección sordo/-sonoro, el tono en caso de voz sonora y los parámetros del filtro $H(z)$: la ganancia G y los coeficientes a_k. [4]

$H(z)$ es un filtro todo polos que modela los efectos de la glotis, el tracto vocal y los labios en la producción de la voz. De esta manera, las contribuciones del modelo (excitación y filtro) están relacionadas en el dominio del tiempo por la siguiente ecuación de convolución:

$$x(n) = u(n) * h(n) \tag{2.21}$$

2.6.2. Extracción de Formantes Mediante LPC

La técnica de predicción lineal consiste en estimar el valor actual de una señal $x(n)$ como una combinación lineal de las muestras anteriores. El valor estimado se

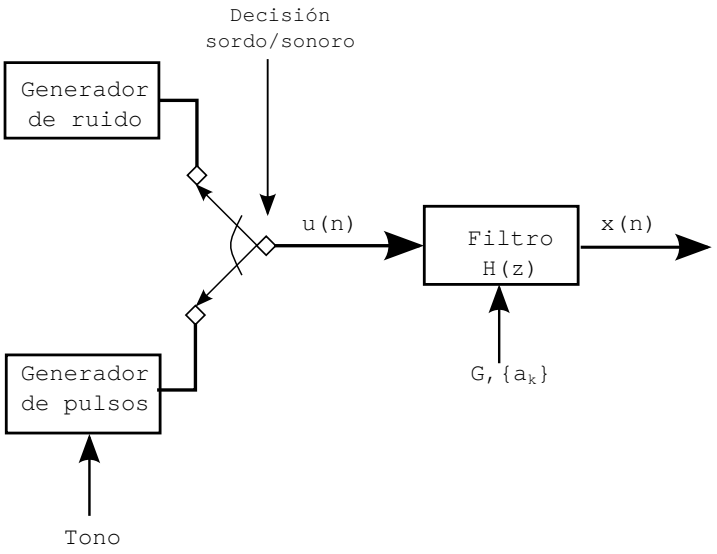

Figura 2.5: Modelo Lineal de Producción de Voz. Fuente [4]

escribe como

$$\hat{x}(n) = -\sum_{k=1}^{N_{LP}} a_k x(n-k), \tag{2.22}$$

donde $N_L P$ es el orden de predicción y a_k son los coeficientes de predicción.

Tomando en cuenta (2.22),la señal de voz puede definirse como:

$$x(n) = -\sum_{k=1}^{N_{LP}} a_k x(n-k) + e(n), \tag{2.23}$$

donde $e(n)$ representa el error de predicción entre el valor real ($x(n)$) y el valor estimado($\hat{x}(n)$) y puede considerarse como la respuesta de un sistema a $x(n)$,

$$E(z) = A(z)X(z), \tag{2.24}$$

donde $A(z)$ es la función de transferencia y viene dada por:

$$A(z) = 1 + \sum_{k=1}^{N_{LP}} a_k z^{-k}, \tag{2.25}$$

Por lo tanto, asumiendo que la señal obedece al modelo (2.5) $A(z)$ será un filtro inverso del filtro $H(z)$.

Siendo $H(z)$ el filtro que modela el tracto vocal, las frecuencias de resonancia de este filtro son los formantes. Gráficamente, los máximos de la respuesta del tracto vocal se corresponden con los formantes. [9] [5] [4]

2.6.3. Extracción de Pitch mediante Cepstrum

Basado en el modelo (2.5), la señal en el dominio del tiempo viene dada por la convolución de ambas contribuciones: la excitación y el tracto vocal. El análisis Cepstrum, es una técnica diseñada para separar estas componentes mediante una transformación de la señal $x(n)$ a un dominio en el que la convolución es una suma. Partiendo de 2.21, se toma la Transformada de Fourier (TF), luego se extrae la magnitud y se transforma a escala logaritmica y se toma la Transformada Inversa de Fourier (TIF). El diagrama de bloques se muestra en la figura(2.6).

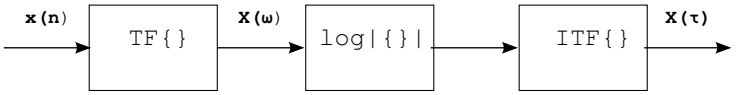

Figura 2.6: Transformación al Dominio Cepstral. Fuente [5]

Esta última transformación (TIF), toma la función de vuelta en el dominio del tiempo, pero no es el mismo de la señal original, de hecho es una medición de la tasa de cambio de las magnitudes espectrales. Este dominio es el llamado Cepstrum y el eje del tiempo se denomina eje quefrency.

La extracción de la frecuencia fundamental de la voz mediante este análisis, consiste en: una vez realizada la transformación, ubicar el máximo pico en el cepstrum de potencia. La posición en el eje quefrency de este máximo, está relacionada con la frecuencia fundamental de la voz mediante la siguiente expresión:

$$f_0 = \frac{fs}{qf_{max}}, \tag{2.26}$$

donde fs es la frecuencia de muestreo en Hertz y qf_{max} es la posición del máximo en el eje quefrency. [5] [31]

2.7. Espectrograma

El espectrograma es una representación de la distribución de energía de una señal en el plano tiempo-frecuencia, como una función que depende de ambos dominios, obtenida mediante la Transformada de Fourier de Tiempo Reducido (STFT), descrita anteriormente.[32]

La gran ventaja del uso de espectrogramas para analizar señales de voz es que esta herramienta permite la representación de la frecuencia de cada componente armónico, la intensidad de cada uno y el instante del fenómeno, todo en el mismo gráfico. El eje de las abscisas corresponde al dominio del tiempo, el eje de las ordenadas corresponde a la frecuencia y la intensidad se muestra mediante escala de grises.

El espectrograma también se ve afectado por el principio de incertidumbre mencionado en secciones anteriores, de esta manera se condiciona la resolución tanto frecuencial como temporal que pueda obtenerse, existiendo así dos tipos de espectrograma: de banda estrecha y de banda ancha, el primero presenta mejor resolución frecuencial mientras que el segundo presenta mejor resolución temporal, la elección de cierto tipo de espectrograma dependerá de la finalidad del análisis realizado. [33].

Los parámetros implicados en el espectrograma son la longitud de ventana, tipo de ventana, tamaño de salto y longitud de la TDF. La eficiencia de los resultados obtenidos dependerá de la elección de estos parámetros al momento de calcular el mismo.[32] [23].

Capítulo III

Procedimientos de la investigación

Con el fin de cumplir con cada objetivo específico planteado en la investigación, y por consiguiente, cumplir el objetivo general; el desarrollo de la investigación fue dividido en cuatro fases, las cuales se detallan a continuación:

3.1. Fase I. Revisión del estado del arte y recopilación de los algoritmos que permitan realizar el cómputo la TDF

Para realizar el análisis de señales, sea cual sea su naturaleza, la obtención de resultados útiles depende en gran manera de establecer una técnica de análisis que permita la determinación de los parámetros que caracterizan al tipo de señal bajo estudio.

Por esta razón, la investigación inició con una revisión de las características de las señales de voz, para así establecer las características espectrales de la voz más relevantes, así como las técnicas para la extracción de las mismas.

En este sentido, se realizó el estudio del modelo de producción de voz humana, concluyendo que los parámetros principales de cualquier señal de voz son: la frecuencia fundamental y los formantes, por lo que resultó de interés su determinación así como también la determinación del espectro en frecuencias.

Asimismo, luego de estudiar las técnicas de extracción de los parámetros espectrales, se determinó que una de las técnicas más eficientes para la estimación de formantes es la basada en predicción lineal y para la extracción de la frecuencia fundamental o pitch, es la técnica de análisis mediante Cepstrum; en ambos procedimientos la extracción se basa en el modelo lineal de producción de voz (2.5) y se encuentran descritos en el capítulo II de este documento.[5]

Por otro lado, debido a la gran utilidad que tiene la Transformada Discreta de Fourier en el análisis de señales, se realizó el estudio de métodos o técnicas para poder realizar el análisis de señales de voz mediante TDF.

Ahora bien, de acuerdo con las características de las señales de voz y en miras de que la interfaz producida de la investigación resultara útil para cualquier tipo de señal de voz hablada, se decidió adoptar la técnica de STFT, para realizar la transformación de los datos al dominio de la frecuencia. La cual, tal como se mencionó en secciones anteriores, no es más que un conjunto de TDF de secciones enventanadas de la señal original que se consideran estacionarias.

Con el uso de la técnica de análisis mencionada, se hace posible la aplicación de algoritmos de Transformada Rápida de Fourier a las señales de voz, por lo que se consultó diferentes fuentes bibliográficas en búsqueda de algoritmos eficientes de TDF procedentes de investigaciones relevantes en el área de Procesamiento Digital de Señales

En general, la fase consistió en revisión de bibliografía relacionada al Procesamiento Digital de Señales orientado al tratamiento de señales de voz; y se planteó de tal manera que al final de la misma, por una parte se pudiera conocer las características espectrales de las voz más relevantes, así como las técnicas para su extracción; y por la otra, establecer la técnica de análisis para las señales de voz mediante algoritmos eficientes de TDF.

3.2. Fase II. Selección de los algoritmos más significativos que permitan realizar el cómputo la TDF y codificación en Python

En esta fase se procedió a realizar la revisión y selección de los algoritmos a ser incluidos en la interfaz, reduciendo el número de algoritmos a los más significativos, para luego evaluar el rendimiento mediante simulaciones de pruebas y codificarlos en Python. También se realizó la codificación de los algoritmos para estimar los parámetros espectrales de las señales de voz.

3.2.1. Selección y codificación de algoritmos de FFT.

La fase consistió en realizar el estudio de las características teóricas, beneficios y desventajas de los algoritmos recopilados en la fase I de la investigación, para realizar una selección preliminar de algoritmos eficientes de TDF, considerando relevancia y eficiencia. En una primera selección se determinó usar los siguientes algoritmos:

1. DIF base-2
2. DIT base-2
3. PFA
4. Goertzel

Fueron escogidos los algoritmos de base 2, porque además de que son eficientes comparados con la aplicación por definición de la TDF, son los algoritmos clásicos de FFT y la mayoría de las aplicaciones prácticas de la TDF están basados en los mismos.

Asimismo, se escogió el Algoritmo de Factores Primos (PFA) el cual es aplicable cuando el número de muestras de la señal de entrada puede descomponerse en

factores relativamente primos, y el algoritmo de Goertzel de segundo orden, que es aplicable a secuencias de cualquier longitud; pues ambos suponen una mejora en eficiencia con respecto a la TDF por definición.

Luego de esta selección preliminar, se realizaron simulaciones de prueba para cada uno de los algoritmos seleccionados, se disponía de códigos desarrollados en MATLAB y Fortran, sin embargo para la aplicación final se decidió emplear Fortran, ya que es un lenguaje de programación que es estándar de elección en la computación científica. Por esta razón fueron empleadas adaptaciones de los algoritmos disponibles en [2], escritos en FORTRAN 77. El procedimiento para las simulaciones realizadas consistió en determinar la TDF de secuencias aleatorias, mediante los algoritmos bajo prueba y comparar los resultados con la TDF determinada mediante el paquete para computación científica con Python, Numpy. [34]

Luego de esto, se realizó un estudio de las posibilidades existentes para codificar los algoritmos en Python: traducción directa del lenguaje original a Python o la creación de módulos de extensión importables a Python.

Una vez hecho esto, se decidió no realizar la traducción desde Fortran a Python, si no que fueron creados módulos de extensión que son importables al código Python. Este es un paradigma de programación bastante utilizado en la actualidad, que consiste en combinar lenguajes de diferentes niveles para obtener los beneficios que brinda cada uno.

Python es un lenguaje de alto nivel que ofrece un entorno de programación interactivo que simplifica y acelera el desarrollo de modelos computacionales pero la velocidad para resolver el modelo puede resultar inaceptable, ya que se interpreta código de bytes de alto nivel y al ejecutar código Python el intérprete invierte la mayoría de su tiempo en averiguar qué operación de bajo nivel hacer y en extraer los datos para esta operación de bajo nivel.

Por esta razón el uso de librerías de bajo nivel conduce a un mejor rendimiento, ya que cuando se ejecuta código en un módulo de extensión, la máquina virtual de Python ya no interpreta código de bytes de alto nivel sino que se ejecuta código

máquina directamente. Esto elimina el gasto de recursos del intérprete, mientras que cualquier operación dentro de ese módulo de extensión se está ejecutando.

Por tanto, se decidió que las operaciones de mayor carga computacional: el cómputo de la FFT, se realizarían con Fortran, que ofrece robustez en lo que a cálculo científico se refiere.

La generación de los módulos de extensión fue realizada mediante una herramienta de extensión de Python denominada f2Py el cual es un generador de interfaz de Fortran a Python. La estrategia utilizada es la denominada vía inteligente y fácil,la cual se encuentra descrita en el apéndice B de este documento. B

En el apéndice A se muestran los códigos de origen Fortran, a partir de los cuales fueron generados los módulos de extensión. Asimismo, con el fin de comparar resultados en lo que a rendimiento se refiere, se decidió incluir el cálculo de la TDF por definición, por lo que fue desarrollado un código en Python para la aplicación de la misma. A

Luego de la creación de las librerías compartidas, se llevó a cabo una nueva etapa de simulaciones, pero esta vez con pequeñas tramas de señales de voz.

Para los algoritmos DIF y DIT, ambos de base 2, se encontró la necesidad de realizar *zero padding*, que consiste en agregar cierto número de muestras de valor cero al segmento que se analizará, esto implica incrementar la longitud aparente de las ventanas de análisis y por lo tanto de la TDF, para así ajustar la longitud de la trama a una potencia entera de dos y poder usar los algoritmos eficientes.

Con respecto al algoritmo de factores primos, se decidió descartarlo de los algoritmos a aplicar en la interfaz final pues su uso para señales de voz e incluso conjuntos de datos de gran longitud, resulta engorroso, pues la condición para que sea aplicable es que todos los factores sean primos entre sí, es decir, no se puede repetir ningún factor. Lo cual en este caso, no siempre se puede cumplir ya que la mayoría de las aplicaciones para señales digitales (por ejemplo para realizar grabación de voz) ajustan el tamaño de la señal a una potencia entera de dos, mediante

zero padding; por esta misma razón casi nunca hay necesidad de usar un algoritmo más específico que los diseñados para N potencia entera de dos. [23]

El algoritmo de Goertzel no presentó ningún detalle adicional que requiriera modificación para su uso, sin embargo para este también se decidió usar *zero padding*, ya que el uso de esta técnica ayuda a definir mejor la forma espectral, pues la resolución en frecuencia se incrementa. [23]

La aplicación directa de la TDF se hizo a través de la interpretación de la misma como un producto matricial. [26]

Los módulos de extensión para cada uno de los algoritmos, incluyendo PFA, se encuentran disponibles en la carpeta `lib_FFT` de la aplicación y pueden ser utilizados en cualquier otro proyecto en Python, mediante el comando `import`.

En resumen, se decidió implementar: DIF base 2, DIT base 2, Goertzel y TDF directa.

3.2.2. Codificación de algoritmos para extracción de parámetros espectrales de las señales de voz.

3.2.2.1. Cómputo de la FFT

En caso de que el análisis sea local (la longitud de la ventana es igual al número de muestras de la señal de entrada), se usa la función `AnalisisLocal` cuyo código en Python se encuentra en el apéndice A, el procedimiento es simplemente enventanar la señal, ajustar la longitud a la siguiente potencia de dos, realizar el *zero padding*, determinar la TDF y calcular los parámetros que hayan sido solicitados por el usuario.

Ahora bien, si el análisis es mediante la técnica STFT se usa la función definida en Python, en la cual se determina el número de tramas a partir de la longitud de la ventana y del parámetro de solapamiento (`noverlap`), luego se procede para cada trama de manera similar a la función `AnalisisLocal`. El código desarrollado para esta función es una adaptación del código MATLAB que se encuentra en [23].

Cabe destacar que el *zero padding* realizado a las señales bajo estudio después de ser enventanadas, se realizó insertando las muestras de valor cero en la mitad de la señal/trama bajo análisis, esto se conoce como *zero padding* de fase cero, el cual es la opción correcta cuando se utilizan ventanas de fase cero como las empleadas en esta investigación. [23].

3.2.2.2. Estimación de formantes

Para la estimación de los formantes se realizó un análisis mediante predicción lineal con una adaptación del procedimiento descrito en [12], fue creada una función en Python que recibe como parámetros de entrada la señal, la longitud de la misma, la frecuencia de muestreo, el número de coeficientes a determinar y el nombre del algoritmo eficiente de TDF. El procedimiento se describe a continuación:

Se realiza el preénfasis, para reducir el rango dinámico del espectro de la señal de voz, esto se hace mediante un filtro pasa altos, cuya función de transferencia es

$$H_p(z) = 1 - \alpha z^{-1}, \tag{3.1}$$

el valor de α usualmente es fijado entre $0,9$ y 1, en este caso, se utilizó $\alpha = 0,9$, este filtro es implementado mediante la función `lfilter` del módulo `Scipy` de Python. Seguidamente se hace la determinación de los coeficientes mediante la función `lpc`, implementada igualmente desde un módulo de Python, la cual estima los coeficientes de predicción lineal usando el método de autocorrelación mediante la recursión de Levinson-Durbin. Para determinar el número de coeficientes se utilizó la siguiente relación

$$N = 2 + \frac{fs}{1000} \tag{3.2}$$

donde fs es la frecuencia de muestreo.

Luego de la determinación de los coeficientes, las frecuencias formantes se estiman con base en la relación entre los formantes y los polos del filtro del tracto vocal $H(z)$ (2.6.2). Recordando que el denominador de la función de transferencia está

relacionado con los coeficientes de predicción lineal, este se factoriza como:

$$1 + \sum_{k=1}^{N_{LP}} a_k z^{-k} = \prod_{i=1}^{N_{LP}} (1 - c_i z^{-1}) \tag{3.3}$$

Donde c_i es un conjunto de números complejos donde cada par de polos conjugados representa una resonancia a la frecuencia:

$$\hat{F}_i = (\frac{fs}{2\pi}) \arctan[\frac{Im(c_i)}{Re(c_i)}] \tag{3.4}$$

Ahora bien, la raíz representa un formante si se cumple la siguiente condición:

$$\sqrt{Im(c_k)^2 + Re(c_k)^2} \geqslant 0,7 \tag{3.5}$$

Para implementar este método en Python, se definieron las funciones `detcoef` y `detformantes`, la primera para realizar la determinación de los coeficientes de predicción lineal y la segunda para realizar la determinación de los formantes haciendo uso de la primera. [5]

Así mismo también se codificó un algoritmo para estimar la envolvente espectral haciendo uso de la función `detcoef` y de la función `Algoritmo`, que determina la FFT mediante los algoritmos eficientes.

3.2.2.3. Estimación de pitch

Para la determinación del pitch se utilizó la técnica del análisis en el dominio Cepstral, para esto se creo una función llamada `Pitchceps`, los parámetros de entrada a la función creada para esta determinación son: la TDF de la señal/trama bajo estudio y la frecuencia de muestreo, luego se realizan las operaciones descritas en (2.6.3). Una vez que se ha tomado la Transformada Inversa de la señal, se determina su magnitud y se procede a buscar el valor máximo en el rango de 70-500 Hz, que se corresponderá con el pitch una vez hecha la transformación desde la escala quefrency a la escala lineal, mediante la relación (2.26).

En el apéndice A se muestran los códigos en Python correspondientes a estas determinaciones.

3.3. Fase III. Diseño de interfaz interactiva para el manejo de los algoritmos

En general, se planteó que el análisis consistiera en cargar una señal de voz, seleccionar una porción, fijar los parámetros de análisis: ventana, longitud de ventana, algoritmo TDF, seleccionar los parámetros que se deseen extraer y la visualización de los resultados.

Con este esquema en mente, se decidió incluir cuatro etapas que son importar y preparar señal para el análisis, parámetros de análisis, parámetros a determinar y resultados.

En esta fase se realizó el diseño y codificación de la interfaz de usuario, bajo lenguaje Python. Para cada una de las secciones que se planteó fueron generados los elementos respectivos (botones, listas desplegables, etc.). Asimismo fue redactado el manual de usuario y se generó un archivo con documentación teórica.

Se hizo una revisión de las librerías disponibles para hacer interfaces gráficas de usuario en Python, entre ellas se escogió como herramienta de desarrollo de interfaz el paquete wxPython.

Lo primero que se hizo fue crear un objeto del tipo `wx.Frame`, que cumpliría la función de ventana principal, sobre este se fueron creando cada una de las etapas de cálculo, por tanto se agregaron los elementos (botones, listas desplegables, etc) y se definieron los eventos asociados a cada una de estas, como sigue:

Para la etapa de importar y preparar la señal:

- Importar archivo de extensión `wav`.
- Reproducir el audio importado.

- Consultar la documentación del software.
- Graficar la señal en el dominio del tiempo a través de la librería de Python `matplotlib`.
- Mostrar la frecuencia de muestreo de la señal cargada.
- Seleccionar una porción de la señal cargada.

En la etapa de parámetros de análisis:

- Selección de algoritmo de TDF.
- Selección de función ventana.
- Selección de tipo de análisis: local o STFT.
- Longitud de la función ventana
- Número de muestras de salto o solapamiento.
- Generar vista previa de la ventana seleccionada.

En la etapa de parámetros a determinar:

- Seleccionar espectro de potencia
- Seleccionar estimación de pitch
- Seleccionar estimación de formantes
- Seleccionar envolvente espectral LPC
- Seleccionar envolvente espectral Cepstrum
- Generar Resultados

Para la sección de resultados se creó otra ventana (`wx.Frame`), en esta ventana se crearon los siguientes elementos con sus respectivos eventos:

- Seleccionar parámetro a graficar
- Redibujar
- Generar reporte de resultados en formato `pdf`, mediante la librería de Python `reportlab`.

3.4. Fase IV. Aplicación del software a muestras de voz

En esta fase se procedió a realizar pruebas con diferentes señales de voz, para determinar el grado de similitud entre los resultados obtenidos usando la interfaz creada y los obtenidos mediante otra aplicación de análisis de señales de voz, para así redactar las conclusiones al respecto.

Capítulo IV

Análisis, interpretación y presentación de los resultados

4.1. Software libre interactivo para análisis espectral de voz

La creación de este software se materializó en la fase III de la investigación, a continuación se describen los detalles de funcionamiento.

4.1.1. Pantalla inicial

La aplicación se inicia con una pantalla de 200x400 pixeles, en la cual se encuentran habilitados los botones para importar un archivo de audio de extensión `.wav`, el botón para acceder a la documentación; y , por supuesto el ícono de cerrar y minimizar la ventana. La imagen de la pantalla principal se muestra en la figura (4.1).

Al presionar el botón Importar, se abre una ventana de diálogo que permite ubicar el archivo e importarlo a la interfaz.

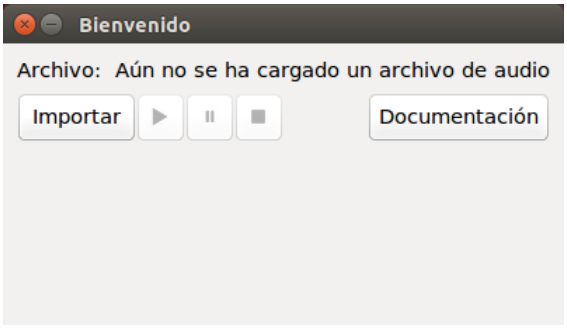

Figura 4.1: Pantalla inicial de la aplicación desarrollada. Fuente: Propia

Si se presiona el botón Documentación se despliega la ventana de diálogo mostrada en la figura (4.2), la cual permite seleccionar el archivo de documentación que se desea consultar, al seleccionar, se abre el archivo de extensión `.pdf` respectivo.

La sección de documentación fue diseñada para servir de orientación al usuario en cuanto al uso de la interfaz y a los conceptos relacionados con el análisis espectral de voz. En consecuencia, fue redactado un manual de usuario y se creó un documento con los aspectos teóricos relevantes para el análisis.

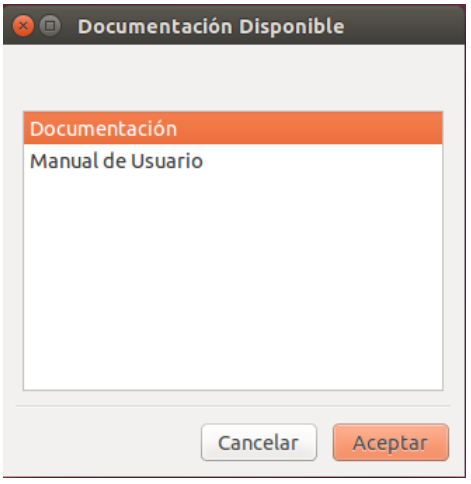

Figura 4.2: Ventana de diálogo que permite seleccionar la documentación a consultar. Fuente: Propia

Si se presiona el ícono Cerrar, aparece un diálogo de confirmación como el mostrado en la figura (4.3) que permite confirmar el cierre de la aplicación.

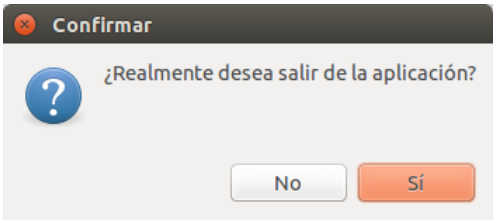

Figura 4.3: Ventana de diálogo que permite confirmar la salida de la aplicación. Fuente: Propia

4.1.2. Ventana Principal

Al importar un archivo de voz, la ventana cambia su tamaño a 800x728 pixeles y su aspecto es como el mostrado en la figura (4.4), en el cual se puede observar las tres etapas del análisis: Preparar la señal a analizar, fijar los parámetros de análisis y características a extraer. A continuación se detallan cada una de estas etapas:

4.1.2.1. Preparación de señal

Esta etapa corresponde a la zona superior izquierda de la ventana principal y se muestra en la figura (4.5). Una vez cargada la señal de audio, en esta etapa surgen los siguientes eventos:

- Se muestra el nombre del archivo en el área superior.
- Se activan los controles de audio: botones Reproducir, Parar y Pausar.
- Se genera la gráfica de la señal en el dominio del tiempo, esta cuenta con una barra de herramientas que permite acercar, alejar y desplazar la escala, también permite guardar la gráfica como una imagen.
- Se carga el valor de la frecuencia de muestreo
- Se activa el área de Selección: se carga la duración de la señal en las entradas de texto Inicio y Fin; y se activan los botones ver y borrar.

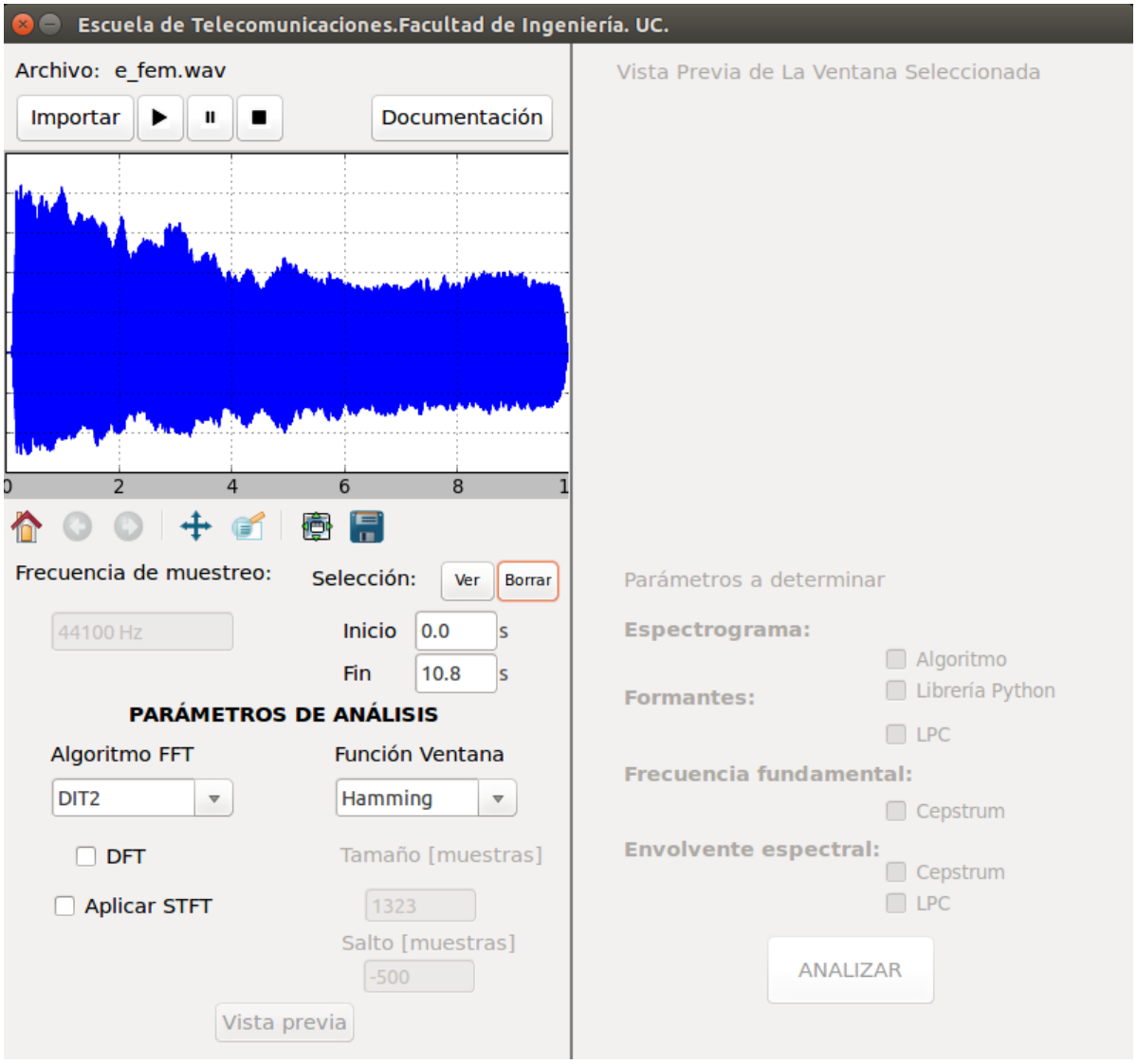

Figura 4.4: Ventana principal de la aplicación desarrollada. Fuente: Propia

El área de selección permite introducir el inicio y fin (en segundos) de la muestra que se desea analizar. Una vez que se han introducido los datos, se debe seleccionar el botón Ver, con esto se sombreará el área seleccionada en la gráfica en el dominio del tiempo y se activará el botón Vista Previa de la siguiente etapa. Si se presiona borrar, se reestablecen los datos.

Figura 4.5: Área de preparación de señal. Fuente: Propia

4.1.2.2. Parámetros de análisis

En esta sección se definen los parámetros de análisis para la aplicación de la TDF a la señal de voz y pertenece al área de interfaz mostrada en la figura (4.6).

Figura 4.6: Área de parámetros de análisis. Fuente: Propia

La forma de la ventana se elige mediante una lista desplegable en la cual se

encuentran disponibles 16 funciones ventana obtenidas mediante la librería de Python, a saber: Bartlett, Bartlett-Hann, Blackman. Blackman-Harris, Bohman, Coseno, Dolph-Chebyshev, Flat Top, Gauss, Hamming, Hann, Kaiser, Nuttall, Parzen y Rectangular. La función ventana por defecto es la ventana de Hamming; en caso de que el análisis sea mediante STFT, la longitud de la ventana es introducida por el usuario mediante una entrada de texto, el valor por defecto es 1323 muestras; y el solapamiento es introducido por el usuario en la entrada de texto Salto, si es un valor negativo es el número de muestras que avanza la ventana, si es un número positivo es el número de muestras que se solapan, el valor por defecto es -500.

Los algoritmos de FFT se encuentran disponibles en una lista desplegable, el usuario puede escoger entre un algoritmo FFT o la aplicación por definición de la TDF.

Una vez que se han establecido los parámetros de análisis, se debe presionar el botón Vista Previa para avanzar a la siguiente etapa. Presionar este botón genera la vista previa de la función ventana seleccionada y activa la sección de Parámetros a Determinar.

4.1.2.3. Parámetros a determinar

En la figura (4.7) se muestra el área de interfaz donde se encuentra esta esta sección, en la cual el usuario puede escoger los parámetros que desea extraer mediante el análisis de la señal de voz. Una vez seleccionados los parámetros, se debe presionar el botón Analizar para generar los resultados.

4.1.2.4. Ventana de Resultados

Esta sección está diseñada para mostrar al usuario los resultados del análisis, asimismo tiene la opción de generar un reporte de resultados. Consiste en una ventana emergente que se genera si y solo si hay al menos un parámetro seleccionado en la sección de Parámetros a Determinar, su aspecto es el mostrado en la figura 4.8.

Figura 4.7: Área de parámetros a determinar. Fuente: Propia

En esta ventana (4.8) el usuario puede seleccionar qué parámetro visualizar, mediante una lista desplegable que muestra los parámetros disponibles. Una vez seleccionado el parámetro, se debe presionar el botón Redibujar para poder visualizarlo. En el caso del espectrograma se brinda al usuario la posibilidad de escoger entre un espectrograma con escala de colores o en escala de grises. En caso de que el usuario escoja redibujar un parámetro que no fue seleccionado en la sección de Parámetros a Determinar, se muestra la ventana de información de la figura (4.9)

El botón Reporte, permite al usuario generar un reporte de resultados en formato pdf, donde se muestran los datos de análisis, la media y moda observada en la determinación de la frecuencia fundamental y los formantes; y el espectrograma en caso de que haya sido calculado. Al presionar este botón, se genera una ventana de entrada de texto que permite al usuario establecer el nombre del archivo a generar, una breve descripción de la señal analizada y el nombre de usuario, una vez que se genera el reporte se muestra al usuario el mensaje de la figura(4.10)

4.2. Aplicación del software a muestras de voz

Se procedió a realizar pruebas de extracción de características con señales de voz producidas de manera natural, que consistieron en la pronunciación de manera sostenida de las cinco vocales, por parte de dos sujetos, uno de sexo masculino y de

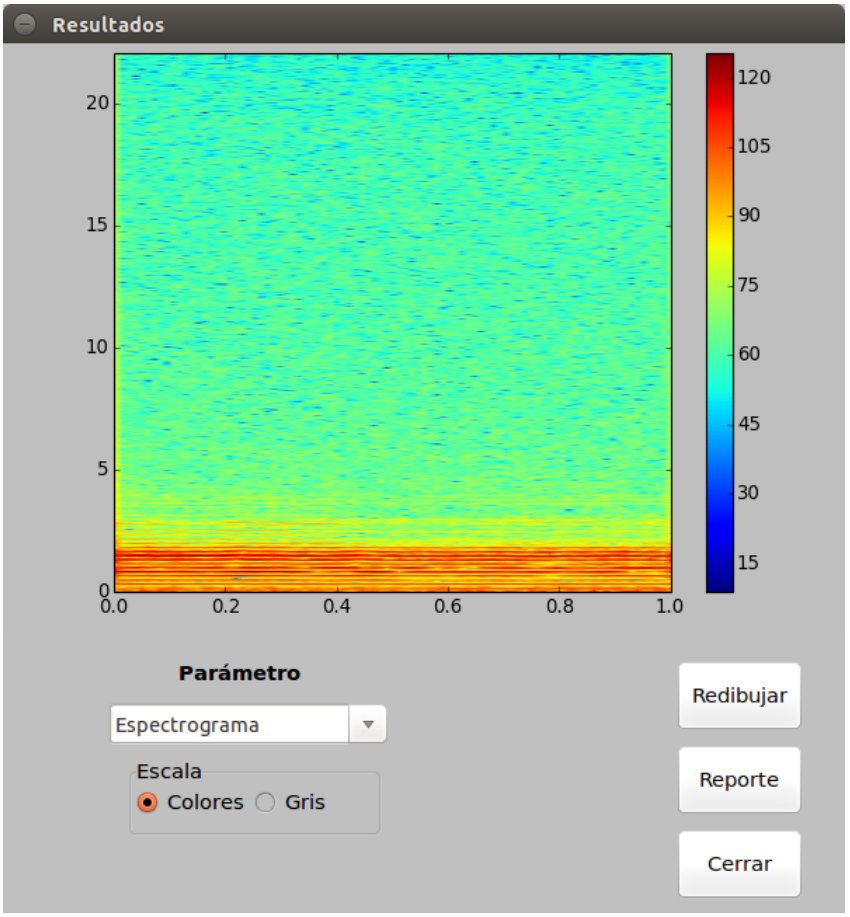

Figura 4.8: Ventana de resultados. Fuente: Propia

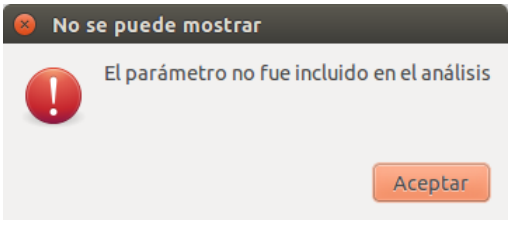

Figura 4.9: Mensaje de error en botón Redibujar. Fuente: Propia

24 años de edad y otro de sexo femenino de 54 años de edad, esto generó un total de 10 señales de voz.

En términos generales, para cada una de las señales el experimento consistió en tomar una muestra de cinco segundos y extraer tanto el pitch como los formantes,

Figura 4.10: Mensaje de generación de reporte. Fuente: Propia

mediante análisis de tiempo reducido, usando como herramienta el software desarrollado y otro software libre para análisis de señales de voz denominado Praat, para luego comparar los resultados. En el caso del análisis en tiempo reducido, se usó Hamming de longitud 1323 muestras (30 ms) como función ventana y avance/salto de 500 muestras y los resultados mostrados corresponden a la media observada en la muestra de tiempo estudiada.

En las secciones siguientes se muestra, mediante tablas, los resultados obtenidos para cada uno de los parámetros, se usó como criterio para realizar la comparación entre los resultados obtenidos, la diferencia porcentual entre las determinaciones, tomando como referencia el resultado obtenido mediante Praat.

4.2.1. Resultados de Extracción de Pitch

Como se ha mencionado en secciones anteriores, en esta investigación la técnica de detección de Pitch empleada es el análisis en dominio Cepstral, los resultados en la extracción de pitch para cada uno de los sujetos considerados en este análisis, se muestra en las tablas (4.1- 4.2).

Se observa que, en el caso del sujeto de sexo masculino (Tabla 4.1), para todas las vocales la diferencia se produce por defecto y en general, se produce una media de diferencia de 5,96 %. Se logra menos de 3 % de diferencia en la determinación del pitch para la emisión de las tres primeras vocales (a,e,i); y para las últimas dos vocales (o,u) la diferencia aumenta. La mínima diferencia se produce en la emisión de la vocal «i», en el caso de la o la diferencia aumenta, pero aún así no supera el

Tabla 4.1: Resultados de determinación de pitch mediante análisis de tiempo reducido para voz de sujeto de sexo masculino de 24 años de edad. Fuente: Propia.

Vocal	Pitch Estimado [Hz]	Pitch (Praat) [Hz]	Dif [%]
a	145,50	148,54	2,04
e	161,49	164,13	1,61
i	218,00	218,30	0,14
o	133,47	148,01	9,82
u	128,93	155,81	16,18

10 %, por lo que se considera dentro del límite de tolerancia; sin embargo en el caso de la vocal «u», la diferencia aumenta casi al doble de este último.

Se decidió variar la longitud de la función ventana a 1764 muestras (40 ms) y avance/salto de 650 muestras, aproximadamente 63 % de solapamiento, para la determinación del pitch en el caso de la emisión de la vocal «u», obteniendo un valor de pitch estimado de 153,21 [Hz], lo que reduce la diferencia de 16,18 % a 0,39 %. De esta manera se evidencia que la variación del tamaño de la ventana influye en la determinación de los resultados.

Tabla 4.2: Resultados de determinación de pitch mediante análisis de tiempo reducido para voz de sujeto de sexo femenino de 54 años de edad. Fuente: Propia

Vocal	Pitch Estimado [Hz]	Pitch (Praat) [Hz]	Dif [%]
a	186,94	187,87	0,49
e	189,82	190,34	0,27
i	213,15	213,71	0,26
o	187,56	201,45	6,89
u	187,01	211,75	11,68

En el caso del sujeto de sexo femenino de 54 años de edad (Tabla 4.2), la media de diferencia general es de 3,92 %, las diferencias mínimas, que no superan el 1 % al igual que en el caso del sujeto de sexo masculino, se obtienen en la estimación de pitch para las tres primeras vocales (a,e,i), produciéndose la diferencia mínima en la señal correspondiente a la emisión de la vocal «i». Las diferencias aumentan para las últimas dos vocales (o,u), nuevamente, el valor de diferencia obtenido en la emisión de la vocal «o» , se encuentra en un rango tolerable. Nuevamente, el valor

de pitch obtenido en la emisión de la vocal «u», presenta mayor diferencia, aunque no tan elevado como en el caso del sujeto de sexo masculino.

Variando la longitud de la ventana, para la determinación del pitch en la emisión de la vocal «u», se obtiene un valor de 202,95 Hz, lo que disminuye la diferencia a 4,16 %. Para esto la longitud de la ventana se incrementó a 2205 muestras y se ajustó el salto a 771 muestras (aproximadamente 65 % de solapamiento).

4.2.2. Resultados en estimación de formantes

4.2.2.1. Primer Formante

Tabla 4.3: Resultados de extracción de primer formante mediante análisis de tiempo reducido para voz de sujeto de sexo masculino de 24 años de edad. Fuente: Propia

Vocal	F1 Estimado [Hz]	F1 (Praat) [Hz]	Dif [%]
a	688,20	708,81	2,91
e	345,03	369,11	6,52
i	357,51	388,84	8,06
o	449,46	484,98	7,32
u	355,04	343,96	-3,22

Tabla 4.4: Resultados de extracción de primer formante mediante análisis de tiempo reducido para voz de sujeto de sexo femenino de 54 años de edad. Fuente: Propia

Vocal	F1 Estimado [Hz]	F1 (Praat) [Hz]	Dif [%]
a	766,05	811,77	5,63
e	420,35	471,10	10,77
i	323,40	383,46	15,66
o	430,85	445,34	3,25
u	360,46	396,77	9,15

En la estimación del primer formante, en el caso del sujeto de sexo masculino se obtuvo una media de diferencia porcentual absoluta de 5,61 % y, en general, la diferencia se encuentra por debajo del 10 %, el mínimo valor de diferencia de estimación se encontró en la vocal «a» y el máximo en la vocal «i». En el caso, del sujeto

de sexo femenino el valor medio de diferencia porcentual absoluto obtenido fue de 8,98 %, obteniendo la mayor diferencia en la estimación del primer formante en la vocal «i».

4.2.2.2. Segundo Formante

Tabla 4.5: Resultados de extracción de segundo formante mediante análisis de tiempo reducido para voz de sujeto de sexo masculino de 24 años de edad. Fuente: Propia

Vocal	F2 Estimado [Hz]	F2 (Praat) [Hz]	Dif [%]
a	1483,49	1473,98	-0,65
e	1154,85	2424,38	52,37
i	1962.04	2348,80	16,47
o	968,32	1026,03	5,62
u	884,17	743,45	-18,93

Tabla 4.6: Resultados de extracción de segundo formante mediante análisis de tiempo reducido para voz de sujeto de sexo femenino de 54 años de edad. Fuente: Propia

Vocal	F2 Estimado [Hz]	F2 (Praat) [Hz]	Dif [%]
a	1380,17	1435,77	3,87
e	1841,94	2287,81	19,49
i	1875,45	2556,17	26,63
o	900,69	844,04	-6,71
u	1209,09	661,44	-82,80

Para la estimación del segundo formante se obtuvo, en el caso del sujeto de sexo masculino, una media de diferencia absoluta de 18,81 %, obteniendo la mayor diferencia en la estimación realizada en la pronunciación de la vocal «e». En el caso del sujeto de sexo femenino se observó una media de diferencia absoluta de 27,90 %, obteniendo la mayor diferencia en el caso de la letra u.

4.2.2.3. Tercer Formante

Tabla 4.7: Resultados de extracción de tercer formante mediante análisis de tiempo reducido para voz de sujeto de sexo masculino de 24 años de edad. Fuente: Propia

Vocal	F3 Estimado [Hz]	F3 (Praat) [Hz]	Dif [%]
a	2446,40	2425,55	-0,86
e	1154,85	3009,02	61,62
i	2649,48	3301,01	19,74
o	2276,98	2478,77	8,14
u	2288,45	2606,05	12,19

Tabla 4.8: Resultados de extracción de tercer formante mediante análisis de tiempo reducido para voz de sujeto de sexo femenino de 54 años de edad. Fuente: Propia.

Vocal	F3 Estimado [Hz]	F3 (Praat) [Hz]	Dif [%]
a	2328,24	2483,68	6,26
e	2619,49	2748,71	4,70
i	2702,79	2803,67	3,60
o	2424,27	2778,01	12,73
u	2513,32	2645,91	5,01

En la estimación del tercer formante, la mayor diferencia se observó es la estimación realizada en la señal correspondiente a la vocal «e» del sujeto de sexo masculino, en cuya señales se obtuvo una media de diferencia absoluta de 20,51 %. En el caso caso del sujeto de sexo femenino se observó una diferencia absoluta en promedio de 6,46 %, observando la mayor diferencia en la vocal «o».

4.2.2.4. Cuarto Formante

Tabla 4.9: Resultados de extracción de cuarto formante mediante análisis de tiempo reducido para voz de sujeto de sexo masculino de 24 años de edad. Fuente: Propia.

Vocal	F4 Estimado [Hz]	F4 (Praat) [Hz]	Dif [%]
a	3665,86	3766,54	2,67
e	3075,41	4046,44	24,00
i	3358,03	4817,64	30,30
o	3359,49	4010,20	16,23
u	3376,34	3447,27	2,06

Tabla 4.10: Resultados de extracción de cuarto formante mediante análisis de tiempo reducido para voz de sujeto de sexo femenino de 54 años de edad. Fuente: Propia.

Vocal	F4 Estimado [Hz]	F4 (Praat) [Hz]	Dif [%]
a	3272,40	3449,28	5,13
e	3498,09	3833,26	8,74
i	3305,33	3805,56	13,14
o	3257,05	4316,98	24,55
u	3596,52	3965,17	9,30

Para el cuarto formante se observó la mayor diferencia porcentual en la vocal «i»,en el caso del sujeto de sexo masculino, y en promedio la diferencia absoluta es de 15,05 %. En el caso del sujeto de sexo femenino la media de diferencia absoluta fue de 12,17 % y la mayor diferencia se observó en la vocal «o».

Para todas las vocales, las mejores estimaciones (aquellas con menor diferencia) fueron la del primer y tercer formante, en el caso del sujeto de sexo femenino y la estimación del primer formante en el caso del sujeto de sexo masculino. Todas con un diferencia absoluta media menor a 10 %.

Tabla 4.11: Diferencia absoluta porcentual en las estimaciones realizadas para el sujeto de sexo masculino. Fuente: Propia.

Vocal	Dif F1 [%]	Dif F2 [%]	Dif F3 [%]	Dif F4 [%]	Media [%]
a	2,91	0,65	0,86	2,67	1,77
e	6,52	52,37	61,62	24,00	36,13
i	8,06	16,47	19,74	30,30	18,64
o	7,32	5,62	8,14	16,23	9,33
u	3,22	18,93	12,19	2,06	9,10
Media	5,61	18,81	20,51	15,05	14,99

Tabla 4.12: Diferencia absoluta porcentual en las estimaciones realizadas para el sujeto de sexo femenino. Fuente: Propia.

Vocal	Dif F1 [%]	Dif F2 [%]	Dif F3 [%]	Dif F4 [%]	Media [%]
a	5,63	3,87	6,26	5,14	5,22
e	10,77	19,49	4,70	8,74	10,93
i	15,66	26,63	3,60	13,14	14,76
o	3,25	6,71	12,73	24,55	11,81
u	9,56	82,77	5,01	9,30	26,66
Media	8,98	27,90	6,46	12,17	13,88

En las tablas (4.11-4.12) se muestran las diferencias absolutas entre las estimaciones realizadas con Praat y las realizadas con el software desarrollado.

Para el sujeto de sexo masculino, los formantes con menor diferencia fueron los obtenidos para las vocales «a»,«o» y «u»; en el caso del sujeto de sexo femenino fueron los obtenidos para las vocales «a» y «e». Las estimaciones con mayor diferencia fueron las realizadas para la vocales «e» y «u», para el sexo masculino y femenino, respectivamente.

Para todos los formantes,las estimaciones con menor diferencia, y por tanto mayor aproximación, fueron las realizadas para la vocal «a»,en ambos sujetos. En el

caso del sujeto de sexo masculino con una diferencia absoluta media de 1,77 % y en el sujeto de sexo femenino 5,22 %.

Se realizó análisis local en la señal de la vocal «u» emitida por el sujeto de sexo femenino, ya que en el segundo formante obtenido para la misma se observó la mayor diferencia porcentual (82,77 %).

En este nuevo análisis la determinación de los formantes se realizó de manera gráfica, a partir de la envolvente espectral determinada mediante LPC. En principio se realizó el análisis a una trama de 1 segundo, cuya envolvente obtenida se muestra en la figura (4.11) y los resultados obtenidos se muestran en la tabla (4.13).

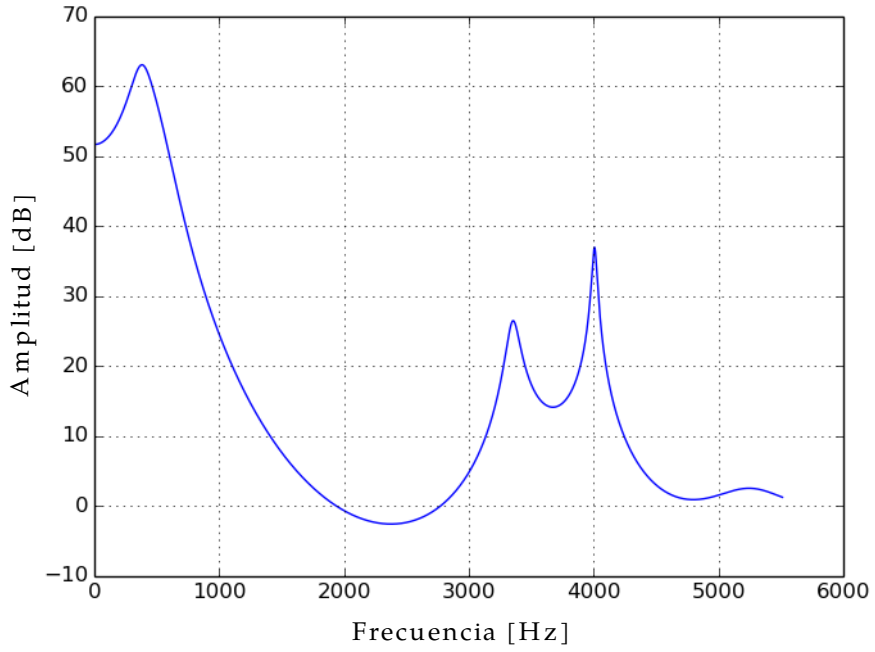

Figura 4.11: Envolvente espectral obtenida mediante LPC, trama 1s. Vocal «u», sujeto de sexo femenino de 54 años de edad. Fuente: Propia

Ahora bien, al observar la figura (4.11) resulta evidente que el segundo formante no es perceptible, esto puede deberse a que la resolución en frecuencia no es lo suficientemente buena como para resolver dos picos tan cercanos,por tanto el

máximo correspondiente al segundo formante está siendo sumado al máximo que corresponde al primer formante.

Sin embargo, si se comparan los valores de diferencia absoluta porcentual obtenidos para cada formante (4.13) con los obtenidos con el análisis anterior 4.12, existe una disminución en la diferencia obtenida en la estimación del primer y cuarto formante.

Tabla 4.13: Resultados de estimación de formantes mediante análisis local a trama de 1s de señal de voz de sujeto de sexo femenino de 54 años de edad. Fuente: Propia.

Técnica	F1 [Hz]	F2 [Hz]	F3 [Hz]	F4 [Hz]
Envolvente	381	0	3595	4007
Praat	393,94	659,76	3359,41	3997,83
|Dif| [%]	3,28	100 %	7,01	0,23

De la misma manera, se realizó la extracción de formantes a partir de la envolvente espectral, pero esta vez con una trama de 30 ms. La envolvente espectral obtenida se muestra en la figura 4.12 y los resultados obtenidos se encuentran en la tabla 4.14.

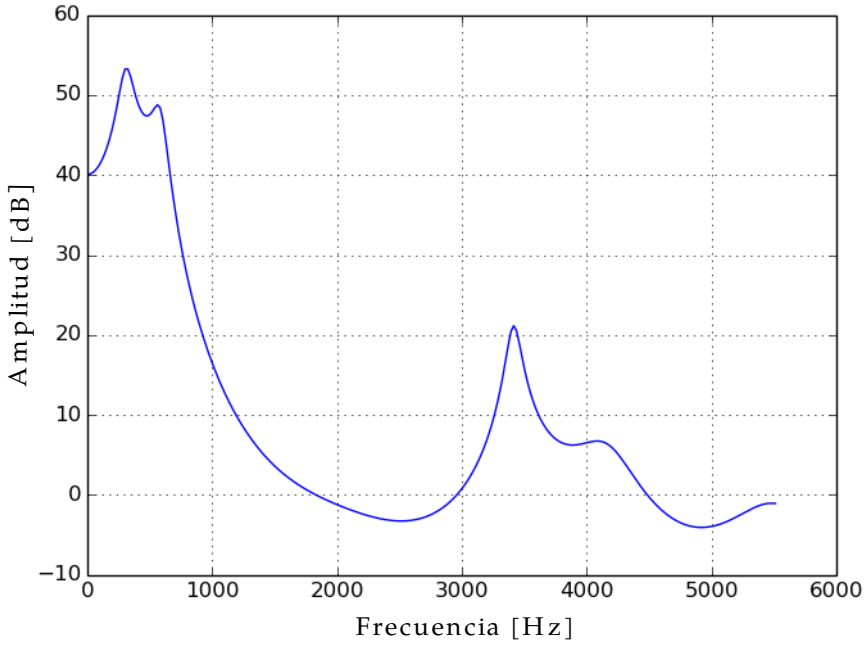

Figura 4.12: Envolvente espectral obtenida mediante LPC, Trama 30ms. Fuente: Propia

Se observa que en esta nueva gráfica (4.12), son perceptibles los cuatro primeros formantes, asimismo es posible estimar el valor del segundo formante con una diferencia porcentual absoluta de 15,34 %. Con este análisis, mejora la estimación el segundo y cuarto formante; la diferencia absoluta para los otros dos formantes aumenta, sin embargo, la media de diferencia absoluta para los formantes estimados es de 11,65 %, lo que implica una disminución de un poco más de la mitad del valor conseguido en el primer análisis 26,66 %.

Tabla 4.14: Resultados de estimación de formantes mediante análisis local a trama de 30ms de señal de voz de sujeto de sexo femenino de 54 años de edad. Fuente: Propia.

Técnica	F1 [Hz]	F2 [Hz]	F3 [Hz] [Hz]	F4 [Hz]
Envolvente	315	560	3400	4100
Praat	396,77	661,44	3125,66	4024,67
|Dif| [%]	20,61	15,34	8,78	1,87

4.2.3. Estimación de parámetros con señales grabadas en ambiente controlado

Se decidió realizar la estimación de parámetros a dos señales de voz, que consistían en la emisión de la vocal «a» sostenida durante cinco segundos, pertenecientes a dos sujetos diferentes, capturadas bajo ambiente controlado y con micrófono de alta directividad, el procedimiento para realizar la estimación fue similar al usado en las pruebas anteriores, a continuación se presentan los resultados obtenidos para cada una de las señales.

Con respecto a la estimación del pitch, los resultados se muestran en la tabla (4.15), donde es evidente que la mayor diferencia absoluta es de $0,12\,\%$

Tabla 4.15: Resultados de estimación de pitch mediante análisis de tiempo reducido a voces bajo ambiente controlado. Fuente: Propia

Sujeto	Pitch Estimado [Hz]	Pitch (Praat) [Hz]	Dif [%]
1	208,46	208,21	-0,12
2	153,53	153,48	0,10

Asimismo, los resultados en extracción de formantes se muestran en las tablas (4.16-4.19).

Tabla 4.16: Resultados de extracción de primer formante mediante análisis de tiempo reducido a voces bajo ambiente controlado. Fuente: Propia

Sujeto	F1 Estimado [Hz]	F1 (Praat) [Hz]	Dif [%]
1	821,01	851,09	3,53
2	601,10	607,02	0,98

Tabla 4.17: Resultados de extracción de segundo formante mediante análisis de tiempo reducido a voces bajo ambiente controlado. Fuente: Propia

Sujeto	F2 Estimado [Hz]	F2 (Praat) [Hz]	Dif [%]
1	1326,38	1343,54	1,28
2	963,25	955,59	-0,80

En general, para todos los formantes extraídos en el sujeto 1, la menor diferencia absoluta es de $1,28\,\%$ y la mayor es $6,47\,\%$, con una media absoluta de $3,48\,\%$ y

Tabla 4.18: Resultados de extracción de tercer formante mediante análisis de tiempo reducido a voces bajo ambiente controlado. Fuente: Propia

Sujeto	F3 Estimado [Hz]	F3 (Praat) [Hz]	Dif [%]
1	2915,37	3116,92	6,47
2	2574,86	2612,59	1,44

Tabla 4.19: Resultados de extracción de cuarto formante mediante análisis de tiempo reducido a voces bajo ambiente controlado. Fuente: Propia

Sujeto	F4 Estimado [Hz]	F4 (Praat) [Hz]	Dif [%]
1	3795,58	3898,12	2,63
2	3126,97	3149,12	0,70

para el sujeto 2, la menor diferencia absoluta obtenida es de $0,70\,\%$ y la mayor es de $1,44\,\%$ con una media de $0,98\,\%$.

Para ambos parámetros, pitch y formantes, es evidente que las diferencias obtenidas son menores comparadas con las estimaciones de los apartados 4.2.1 y 4.2.2, por tanto las condiciones de captura de la señal de voz también influyen en los resultados, por esta razón, en el apéndice D, se incluyen recomendaciones para grabar las señales de voz.

Capítulo V

Conclusiones y recomendaciones

5.1. Conclusiones

El aporte principal de esta investigación fue la creación de un software libre interactivo, bajo lenguaje Python, a través de la librería wxPython, el mismo permite realizar el análisis de señales de voz pregrabadas, mediante análisis local o en tiempo reducido, usando Transformada Discreta de Fourier; diseñado para que el usuario tenga la libertad de fijar los parámetros implicados en el análisis de las señales de voz, con la posibilidad de estimar el pitch, los formantes, espectrograma, y envolvente de la señal.

Con base en la experiencia obtenida en el desarrollo de cada una de las etapas de la investigación se concluye que:

1. Las características esenciales de las señales de voz son: la frecuencia fundamental y los formantes; que pueden ser estimadas a partir de distintas técnicas de extracción, de las cuales resultaron relevantes el análisis en dominio cepstral, para la estimación de la frecuencia fundamental y el análisis de predicción lineal para los formantes.

2. La Transformada Discreta de Fourier es un herramienta de análisis altamente eficiente en el tratamiento de señales de voz, para el cómputo de la misma

existen numerosos algoritmos eficientes, cuya esencia es idéntica: dividir el problema original en subproblemas de menor magnitud, sacando provecho de las características de periodicidad y simetría de la TDF, por esta razón se les denomina de manera común Transformada Rápida de Fourier.

3. Los algoritmos más significativos de FFT, son los algoritmos de base dos en sus versiones Diezmado en Frecuencia (DIF) y Diezmado en Tiempo (DIT), ambos diseñados para cuando el número de muestras de la entrada es una potencia entera de dos; estos son los algoritmos clásicos de FFT por lo que la mayoría de las aplicaciones prácticas están basadas en los mismos, cuya ventaja principal es la disminución del número de operaciones de N^2 a $N\log_2 N$.

4. La implementación de los algoritmos de FFT para realizar el análisis espectral de voz es factible mediante el uso de técnicas que consideren la característica de no estacionariedad de las señales de voz, esto es realizar el análisis en tramas de tiempo reducido en un rango de 10-40 ms, intervalo en el que se considera que las características de la voz no varían de manera considerable.

5. Se comprobó que la variación de los parámetros de análisis: función ventana, longitud y solapamiento, influye en los resultados obtenidos en la estimación de las características esenciales de las señales de voz.

6. Con respecto a la estimación del pitch, se obtuvieron estimaciones con un mínimo margen de diferencia con respecto a otra aplicación de código libre desarrollada con el mismo fin.

7. Con respecto a la extracción de formantes, se comprobó que la mejor estimación se realiza a través de la inspección visual de la envolvente espectral de la señal de voz.

8. Se comprobó que *Python* es un lenguaje de programación altamente extendible, que permite la ejecución de módulos de otros lenguajes como C,C++ y *Fortran*, lo que facilitó la combinación de dos lenguajes: *Fortran* y *Python*, tomando ventaja de los beneficios de cada uno. Debido a esta cualidad, existen diversas herramientas para crear la conexión entre el lenguaje de origen y *Python*, como la utilizada en esta investigación: `F2Py`.

9. Se generó una librería con cuatro algoritmos de FFT disponibles en la carpeta `lib_FFT` del sofware desarrollado, estos módulos son importables a Python y utilizables en cualquier proyecto que requiera cómputo de FFT.

5.2. Recomendaciones

Con base al estudio realizado y a los resultados obtenidos, como mejoras a la presente investigación, se recomienda:

- Añadir un módulo de grabación de voz, que permita realizar esta función a distintas frecuencias de muestreo.
- Optimizar el algoritmo de extracción de formantes
- Incluir la estimación de los parámetros de perturbación de la voz, así como aquellos que determinan la calidad de la misma.

Asimismo, se recomiendan otros estudios que pueden resultar de utilidad en el análisis de señales de voz:

- Realizar estudio comparativo de técnicas para extracción de pitch.
- Realizar estudio comparativo de métodos para la estimación de los coeficientes de predicción lineal, con el objetivo de realizar extracción de formantes y envolvente espectral LPC.
- Estudio de algoritmos para realizar contorno de pitch y seguimiento de formantes.
- Análisis de señales de voz mediante Transformada de Wavelet, así como también el estudio comparativo de rendimiento entre ésta y la Transformada de Fourier.

Apéndice A

Códigos de Algoritmos para el Cómputo de la TDF

1.1. TDF por definición

```
# -*- coding: utf-8 -*-

# Transformada Discreta de Fourier por definicion

import numpy as np
import math
import cmath
def dft(sig):
    N=len(sig)
    y=np.matrix([sig]).T
    # n: base de tiempo discreto
    # k: base de frecuencia discreta
    n = np.matrix([list(range(N))]).T
    k = np.matrix([list(range(N))])
    WN=cmath.exp(-1j*(2*math.pi/N))
    v=n*k
    p=np.array([v])
    r=WN**(p)
    W= np.asmatrix(r)
```

```
    xk=W*y
    xk = np.array(xk)
    xk = xk.reshape(N)
    return xk
```

dft.py

1.2. Algoritmos para N potencia entera de 2

1.2.1. Diezmado en frecuencia de base 2

```
! ALGORITMO DIF DE RAIZ 2
! ORIGINAL DESARROLLADO POR: S. BURRUS, RICE UNIVERSITY, SEPT 1983
! X: PARTE REAL DE LA ENTRADA/SALIDA
! Y: PARTE IMAGINARIA DE LA ENTRADA/SALIDA
! N: NRO DE MUESTRAS DE LA ENTRADA
! M: LOGARITMO BASE 2 DE N.
!-------------------------------------------------------
      SUBROUTINE FFT (X,Y,N,M)
      REAL X(N), Y(N)
      INTEGER N, M
!f2py intent(in) X,Y,N,M
!f2py intent(out) X,Y
!f2py depend(X) N
!f2py depend(M) N
!--------------FFT-----------------------------
!
       N2 = N
       DO 10 K=1, M
            N1 = N2
            N2 = N2/2
            E  = 6.283185307179586/N1
            A  = 0
            DO 20 J = 1, N2
                C = COS(A)
                S = SIN(A)
                A = J*E
                DO 30 I = J, N, N1
```

```
                    L    = I + N2
                    XT   = X(I) - X(L)
                    X(I)    = X(I) + X(L)
                    YT   = Y(I) - Y(L)
                    Y(I) = Y(I) + Y(L)
                    X(L) = C*XT + S*YT
                    Y(L) = C*YT - S*XT
30                    CONTINUE
20                 CONTINUE
10              CONTINUE
!
!------------ORDENAMIENTO-----------------
        J = 1
        N1 = N - 1
        DO 104 I=1, N1
        IF (I.GE.J) GOTO 101
        XT = X(J)
        X(J) = X(I)
        X(I) = XT
        XT   = Y(J)
        Y(J) = Y(I)
        Y(I) = XT
  101       K = N/2
  102       IF (K.GE.J) GOTO 103
        J = J - K
        K = K/2
        GOTO 102
  103       J = J + K
  104   CONTINUE
      RETURN
      END
```

dif2.f

1.2.2. Diezmado en tiempo de base 2

```
! ALGORITMO DIT DE RAIZ 2
! ORIGINAL DESARROLLADO POR: S. BURRUS, RICE UNIVERSITY, SEPT 1985
! X: PARTE REAL DE LA ENTRADA/SALIDA
! Y: PARTE IMAGINARIA DE LA ENTRADA/SALIDA
! N: NRO DE MUESTRAS DE LA ENTRADA
! M: LOGARITMO BASE 2 DE N.
!-------------------------------------------------------
      SUBROUTINE FFT (X,Y,N,M)
      REAL X(N), Y(N)
      INTEGER N,M
!f2py intent(in) X,Y,N,M
!f2py intent(out) X,Y
!f2py depend(X) N
!f2py depend(M) N
!------------ORDEN DE INVERSION DE BITS-----------------
!
        J = 1
        N1 = N - 1
        DO 104 I=1, N1
            IF (I.GE.J) GOTO 101
                XT = X(J)
                X(J) = X(I)
                X(I) = XT
                XT   = Y(J)
                Y(J) = Y(I)
                Y(I) = XT
101             K = N/2
102             IF (K.GE.J) GOTO 103
            J = J - K
            K = K/2
            GOTO 102
103             J = J + K
104         CONTINUE
!--------------FFT-----------------------------
!
        N2 = 1
        DO 10 K = 1, M
            E  = 6.283185307179586/(2*N2)
```

```
            A  = 0
            DO 20 J = 1, N2
            C = COS (A)
            S = SIN (A)
            A = J*E
            DO 30 I = J, N, 2*N2
                        L = I + N2
                        XT = C*X(L) + S*Y(L)
                        YT = C*Y(L) - S*X(L)
                        X(L) = X(I) - XT
                        X(I) = X(I) + XT
                        Y(L) = Y(I) - YT
                        Y(I) = Y(I) + YT
30                  CONTINUE
20              CONTINUE
            N2 = N2+N2
10          CONTINUE
!
        RETURN
        END
```

dit2.f

1.2.3. Goertzel de segundo grado

```
! ALGORITMO DE GOERTZEL DE SEGUNDO ORDEN
! ORIGINAL DESARROLLADO POR: S. BURRUS, RICE UNIVERSITY, SEPT 1983
! X: PARTE REAL DE LA ENTRADA
! Y: PARTE IMAGINARIA DE LA ENTRADA
! N: NRO DE MUESTRAS DE LA ENTRADA
! A: PARTE REAL DE LA SALIDA
! B: PARTE IMAGINARIA DE LA SALIDA
!-------------------------------------------------------
      SUBROUTINE DFT(X,Y,N,A,B)
      real X(N), Y(N), A(N), B(N)
!f2py intent(in) X,Y,N
!f2py intent(out) A,B
!f2py depend(N) X
      Q = 6.283185307179586/N
```

```
      DO  J=1, N
        C  = cos(Q*(J-1))
        S  = sin(Q*(J-1))
        CC = 2*C
        A2 = 0
        B2 = 0
        A1 = X(1)
        B1 = Y(1)
        DO I = 2, N
            T  = A1
            A1 = CC*A1 - A2 + X(I)
            A2 = T
            T  = B1
            B1 = CC*B1 - B2 + Y(I)
            B2 = T
        ENDDO
        A(J) = C*A1 - A2 - S*B1
        B(J) = C*B1 - B2 + S*A1
      ENDDO
!
      RETURN
!
      END
```

goertzel.f

1.2.4. Algoritmo de Factores Primos

```
! ALGORITMO DE FACTORES PRIMOS
! ORIGINAL DESARROLLADO POR: S. BURRUS, RICE UNIVERSITY, SEPT 1983
! X: PARTE REAL DE LA ENTRADA/SALIDA
! Y: PARTE IMAGINARIA DE LA ENTRADA/SALIDA
! N: NRO DE MUESTRAS DE LA ENTRADA
! M: NRO DE FACTORES PRIMOS DE N
! NI: FACTORES PRIMOS DE N
!       N = NI(1)*NI(2)*...*NI(M)
! Tiene modulos para NI = 2,3,4,5,7,8,9,16
!-----------------------------------------------------
!
```

```
      SUBROUTINE PFA(X,Y,N,M,NI)
      INTEGER  NI(M), I(16), IP(16), LP(16)
      REAL X(N), Y(N)
      DATA  C31, C32  / -0.86602540,-1.50000000 /
      DATA  C51, C52  /  0.95105652,-1.53884180 /
      DATA  C53, C54  / -0.36327126, 0.55901699 /
      DATA  C55       / -1.25  /
      DATA  C71, C72  / -1.16666667,-0.79015647 /
      DATA  C73, C74  /  0.055854267, 0.7343022 /
      DATA  C75, C76  /  0.44095855,-0.34087293 /
      DATA  C77, C78  /  0.53396936, 0.87484229 /
      DATA  C81       /  0.70710678 /
      DATA  C95       / -0.50000000 /
      DATA  C92, C93  /  0.93969262, -0.17364818 /
      DATA  C94, C96  /  0.76604444, -0.34202014 /
      DATA  C97, C98  / -0.98480775, -0.64278761 /
      DATA  C162,C163 /  0.38268343,  1.30656297 /
      DATA  C164,C165 /  0.54119610,  0.92387953 /

!f2py intent(in,out) X,Y
!f2py intent(in) N,M,NI
!f2py depend(N) X
!f2py depend(M) NI
!
!-----------------NESTED LOOPS---------------------------------
!
      DO 10 K=1, M
      N1 = NI(K)
      N2 = N/N1
      L  = 1
      N3 = N2 - N1*(N2/N1)
      DO 15 J = 1, N1
      LP(J) = L
      L = L + N3
      IF (L.GT.N1) L = L - N1
15    CONTINUE
!
      DO 20 J=1, N, N1
      IT   = J
      DO 30 L=1, N1
```

```
        I(L) = IT
        IP(LP(L)) = IT
        IT = IT + N2
        IF (IT.GT.N)  IT = IT - N
30         CONTINUE
        GOTO (20,102,103,104,105,20,107,108,109,
     +             20,20,20,20,20,20,116), N1
```

PFA.f

Apéndice B

Módulos de extensión de FORTRAN a Python con f2py

F2py es un generador de interfaz de Fortran a Python, que proporciona una manera simple de importar código Fortran en Python. Actualmente proporciona soporte completo para manejar códigos FORTRAN 77 y soporte parcial para Fortran 90 o códigos más nuevos.

Para generar el código dentro de un módulo importable a Python, se ejecuta el siguiente comando, el cual creará la biblioteca compartida (extensión `.so` en linux):

```
f2py -c nombrearchivo.f -m nombremodulo
```

Donde la opción `-c` le da instrucción a f2py para construir la biblioteca del módulo de extensión que pueda ser importado a Python (sin la opción `-c` podría generar solo las fuentes del módulo de extensión).

La opción `nombrearchivo` es el nombre del archivo de origen que contiene el código Fortran; y la opción `-m nombremodulo` es el nombre del módulo de extensión que será generado.

El resultado generado con f2py se documenta automáticamente y la interfaz es bastante simplificada. La conexión entre Fortran y Python es lograda vía C, por lo que al usar f2py, se autogeneran módulos C los cuales saben cómo usar código Fortran.

Ahora bien, existen tres maneras de utilizar f2py, las cuales se detallan a continuación:

2.1. Vía Fácil

Consiste simplemente en ejecutar la línea de comando antes indicada sin hacer ninguna modificación al código Fortran. Significa dejar que f2py haga todo el trabajo:

- Escanear la información (signature) desde las fuentes Fortran.
- Crear firmas de interfaz para los procedimientos, módulos y colecciones de datos de Fortran.
- Generar las fuentes del módulo contenedor que tienen las funciones de envoltorio necesarias
- Compilar los archivos de origen C y Fortran.
- Construir el módulo envoltorio de Python que puede ser usado inmediatamente para llamar procedimientos Fortran desde Python.

2.2. Vía inteligente

En esta el usuario especifica las firmas de interfaz y f2py hace el resto de las tareas mencionadas anteriormente, el procedimiento a seguir se detalla a continuación:

- Se crea una interfaz de módulo envoltorio inicial con la siguiente línea de comando

```
f2py -m <nombre_modulo> <archivo_fortran> -h <nombre_modulo>.pyf
```

  Este comando escaneará los archivos origen de fortran para firmas de procedimiento y guardará la información obtenida en un archivo de firma (`.pyf`)

- Se revisa el archivo generado y manualmente se inserta la intención de los argumentos con el atributo `intent`, se provee valores por defecto para argumentos opcionales, etc.

- Se construye el módulo envoltorio

```
f2py -c <nombre_modulo>.pyf <archivo_fortran>
```

  Se verifica la documentación de las funciones generadas y se comprueba su funcionalidad. En caso de necesitar alguna mejora, se vuelve al paso 2.

2.3. Vía fácil e inteligente

Esta es una combinación de las vías anteriores, consiste en insertar la información de firma adicional (como por ejemplo la intención de los argumentos con el atributo `intent`) en el código Fortran de origen usando comentarios de f2py (`!f2py`), para luego ejecutar la línea de comando especificada al principio de este apéndice. B

La técnica por defecto es la fácil, la ventaja es que los envoltorios se pueden crear con un solo comando y proporcionan acceso inmediato al código Fortran desde Python, la desventaja es que el usuario debe ser consciente de la posible necesidad de hacer frente a las formas diferentes de ordenamiento de los datos en FORTRAN y C, así como la forma de preparar arreglos en Python para que pasen de manera

eficiente a Fortran y se llenen con los resultados calculados. Estas cosas normalmente surgen por la falta de información de la intención de los argumentos en los procedimientos de FORTRAN 77. Como resultado, f2py debe usar el supuesto más conservador: todos los argumentos son solo entrada. [34]

Apéndice C

Códigos usados para el análisis de señales de voz

3.1. Cómputo de la TDF

```
    def Algoritmo(self,nombre_alg,sig):
        N=len(sig)
        M=int(np.log2(N))
        sig_trans=np.zeros(N,complex)
        if nombre_alg=="DIT2":
            sig_trans.real, sig_trans.imag = dit2.fft(sig.real,sig.imag,M)
        if nombre_alg=="DIF2" :
           sig_trans.real, sig_trans.imag = dif2.fft(sig.real,sig.imag,M)
        if nombre_alg=="Goertzel" :
           sig_trans.real, sig_trans.imag = goertzel.dft(sig.real,sig.imag)
        if nombre_alg=="DFT" :
           sig_trans = dft(sig)
        return sig_trans
```

Interfaz.py

3.2. Análisis LPC

```
    def det_coef(self,x,fs):
        ncoef = 2 + np.round(fs/1000)
        x1 = lfilter([1., -0.90],[1.], x)
        A,e, k  = lpc(x1, ncoef)
        return A
#####################################################
    def Env_LPC(self,xw,nfft,fs, algoritmo):
        coef = self.det_coef(xw,fs)
        coef = np.append(coef,np.zeros(nfft-len(coef)))
        coef_t = self.Algoritmo(algoritmo, coef)
        envolpc = 20*np.log10(1/abs(coef_t))
        return envolpc
#############################################################
    def det_formantes(self,x1,Fs):
        import math
        f_form = np.array([])
        f_form0 = np.array([])
        Coe = self.det_coef(x1,Fs)
        rts = np.roots(Coe)
        rts = [r for r in rts if np.imag(r) >= 0]
        for i in range(len(rts)):
            if abs(rts[i]) >= 0.7 :
                angz = np.arctan2(np.imag(rts[i]), np.real(rts[i]))
                f = angz * (Fs / (2 * math.pi))
                f_form0 = np.append(f_form0,f)
        f_form = sorted(f_form0)
        if f_form[0] > 200 :
            return f_form[0:4]
        else :
            return f_form[1:5]
```

Interfaz.py

3.3. Análisis cepstrum

```
def BuscarPitch(self,graf,fs):
   ini = fs/500
   fin = fs/70
   Max=ini+np.argmax(graf[ini:fin])
   pitch_int = fs/Max
   return pitch_int

def Pitch_ceps(self,xw,fs) :
        mod_xw_T = abs(xw)
        mod_log = 20*np.log10(mod_xw_T)
        mod_lof = np.fft.ifft(mod_log)
        graf = abs(mod_lof)
        pitch_trama_int = self.BuscarPitch(graf,fs)
        return pitch_trama_int

def Env_ceps(self,sspec,Nfft,alg,fs):
    dbsspecfull = 20*np.log10(abs(sspec))
    rcep = np.fft.ifft(dbsspecfull)
    f0 = self.BuscarPitch(abs(rcep),fs)
    periodo = np.round(fs/f0)
    rcep = np.real(rcep)
    nw = 2*periodo-4
    if np.floor(nw/2) == nw/2:
        nw=nw - 1
    w = boxcar(nw)
    wzp = w[(nw/2):nw]
    wzp = np.append(wzp,np.zeros(Nfft-nw))
    wzp = np.append(wzp,w[0:(nw)/2])
    wrcep = wzp*rcep
    rcepenv = self.Algoritmo(alg,wrcep)
    envoceps = np.real(rcepenv)
    envoceps = envoceps - np.mean(envoceps)
    return envoceps
```

Interfaz.py

3.4. Análisis Local

```
                b_envcep, b_envlpc, b_espa, b_espp) :
        xw = signal * ventana
        N = self.nextpow2(len(signal))
        zp = np.zeros(N-len(signal))
        Modd = np.mod(N,2) # 0 si M es par, 1 si es impar
        Mo2 = (N-Modd)/2
        x = np.array(xw[Mo2:N],dtype = float)
        x = np.append(x,zp)
        x = np.append(x,xw[0:Mo2])
        Cesp_env = Lpc_env = np.zeros(N)
        form_14 = np.zeros(4)
        f0 = 0
        espa = 0
        espp = 0
        x_T = self.Algoritmo(algoritmo,x)
        if b_pitch == 1 :
            f0 = self.Pitch_ceps(x_T,fs)
        if b_formante == 1:
            form_14 = self.det_formantes(x,fs)
        if b_envcep == 1 :
            Cesp_env = self.Env_ceps(x_T,N,algoritmo,fs)
        if b_envlpc == 1 :
            Lpc_env = self.Env_LPC(xw,N,fs,algoritmo)
        if b_espa == 1 :
            def log_seg(x,minval=0.0000000001):
                return np.log10(x.clip(min=minval))
            espa = 10*log_seg(abs(x_T))
        if b_espp ==1 :
            def log_seg(x,minval=0.00001):
                return np.log10(x.clip(min=minval))
            espp = 20*log_seg(abs(x_T))
        return x_T, f0, form_14, Cesp_env, Lpc_env, espa, espp
```

Interfaz.py

3.5. STFT

```
    def STFT(self,signal,fs,window,M,noverlap = -500,b_espectro = True,b_pitch =
True,b_Form=True,Alg = 'DIT2'):
        signal = signal[:]
        #Zero padd
        if len(signal) < M :
            signal = np.append(signal,np.zeros(M-len(signal)))

        Modd = np.mod(M,2) # 0 si M es par, 1 si es impar
        Mo2 = (M-Modd)/2
        w = window[:] #Es una columna
        if noverlap<0 :
            nhop = - noverlap
            noverlap = M-nhop
        else :
            nhop = M-noverlap
        nx = len(signal)
        nframes = 1+int(np.ceil(nx/nhop))
        nfft=int(self.nextpow2(M))
        X = np.zeros((nfft,nframes), dtype = complex)
        zp = np.zeros(nfft-M) #Ceros para cada FFT
        xoff = 0 - Mo2
        xframe = np.zeros(M)
        ###Para pitch
        pitch_st=np.zeros(nframes)
        ###Para formantes
        F = np.zeros((4,nframes))
        ########################
        if b_pitch==True or b_Form == True or b_espectro == True :
            for m in range(nframes) :
                if xoff<0 :
                    xframe[0:xoff+M] = signal[0:xoff+M]
                else :
                      if xoff+M > nx :
                          xframe = signal[xoff:nx]
                          xframe = np.append(xframe,np.zeros(xoff+ M-nx))
                      else :
                          xframe = signal[xoff:xoff+M]
```

```
            xw = w * xframe
            xwzp = np.array(xw[Mo2:M],dtype = float)
            xwzp = np.append(xwzp,zp)
            xwzp = np.append(xwzp,xw[0:Mo2])
            xwzp_T = self.Algoritmo(Alg,xwzp)
            if b_pitch==True  :
                pitch_st[m] = self.Pitch_ceps(xwzp_T,fs)
            if b_Form == True :
                F[:,m] = self.det_formantes(xwzp,fs)
            if b_espectro ==True :
                X[:,m] = xwzp_T
            xoff = xoff + nhop
        return X, nframes, nhop, pitch_st, F, nfft
```

Interfaz.py

Apéndice D

Recomendaciones para grabar señales de voz

Para la extracción de los parámetros de la voz, resulta útil la grabación de la vocal a sostenida durante 5-10 segundos, considerando que:

- La intensidad y el tono deben ser los habituales.
- El inicio de la fonación debe ser normal.
- La distancia labios-micrófono debe ser fija, alrededor de 30cm desde la boca.

Asimismo, es conveniente el uso de muestras de voz de alta calidad. Para esto, de acuerdo con [36], [19], [35], [37] y [38], se recomienda:

4.1. Micrófono de alta calidad

Con el uso de micrófonos convencionales para computadora existe la desventaja de captar ruido electrónico proveniente de la misma, por esta razón se recomienda el uso de un micrófono externo de condensador y *tipo cardioide*. Un micrófono cardioide permite captar la fuente sonora disminuyendo el ruido ambiente y la reverberación del sitio donde se realiza la captura de la muestra de voz, ya que capta

mejor el sonido dentro de un área cónica definida desde su frontal; fuera de esta, la sensibilidad del mismo se reduce.

Además, el micrófono debe proveer un factor de distorsión bajo y una gama de frecuencias amplia que corresponda al espectro típico de los sonidos del habla y de cualquier habla atípica que pueda caracterizar a las disfonías. La gama de frecuencias ideal va de 0-20 kHz.

Asimismo, es recomendable ubicar el micrófono en un boom o en cualquier otro soporte colgante en vez de un soporte de mesa, esto ayuda a aislar los ruidos que pueden producirse si el sujeto golpea o patea la mesa, así como para que la posición del micrófono sea fija durante la grabación. Una opción es utilizar micrófonos de diadema con el cual se puede mantener fija la posición, independientemente de los movimientos del sujeto.

Por otro lado, la entrada de la voz debe ser mediante una tarjeta de sonido de entrada balanceada con un preamplificador de bajo ruido.

4.2. Postura adecuada del sujeto

Un cuerpo bien alineado ayuda a reducir la tensión y producir una respiración fácil, factores implicados en la producción de voz, por tanto durante la fonación es conveniente que el sujeto mantenga un equilibrio muscular y estático que permita conseguir la voz con el menor esfuerzo.

El óptimo alineamiento se logra manteniendo el cuerpo en su verticalidad correcta, esto es: alargar la columna vertebral, ensanchar del pecho y equilibrar el peso del cuerpo, asimismo la tensión no debe ser excesiva, se debe mantener una postura correcta de la laringe en el cuello y el cuello estirado pero sin tensión muscular.

4.3. Ambiente controlado

Con respecto al entorno de grabación, debe ser acústicamente preparado sin llegar a los extremos de una cámara anecoica, siempre que sea posible se debe usar una cabina audiométrica.

En caso de no disponer de una cabina audiométrica, el lugar de grabación debe ser lo más silencioso posible, evitando espacios reverberantes pues las reflexiones acústicas pueden producir distorsiones en la señal de audio. Asimismo se debe apagar todos los dispositivos no implicados en la grabación que puedan ser fuente de ruido externo.

Referencias Bibliográficas

[1] Chang, Wei Hsin: *On the Fixed-Point Analysis and Architecture Design of FFT Algorithms*. Tesis de Doctorado, University of California, 2007.

[2] Burrus, C, F Matteo y otros: *Fast Fourier Transforms*. CONNEXIONS. Rice University., 2012.

[3] Oppenheim, Alan. V., Ronald W. Schafer y John R. Buck: *Tratamiento de Señales en Tiempo Discreto.* Pearson Educación,S.A., 2ª edición, 2000.

[4] Hernando, Francisco Javier: *Técnicas de procesado y representación de la señal de voz para el reconocimiento del habla en ambientes ruidosos.* Universidad Politécnica de Cataluña, 1993.

[5] Kammoun, Med, Dorra Gargouri, Mondher Frikha y Ahmed Ben Hamida: *Cepstrum vs. LPC: A Comparative Study for Speech Formant Frequencies Estimation*. GESTS, 19(1), 2006.

[6] Moran, Rosalyn J., Richard B. Reilly, Philip de Chazal y Peter D. Lacy: *Telephony-Based Voice Pathology Assessment Using Automated Speech Analysis*. IEEE TRANSACTIONS ON BIOMEDICAL ENGINEERING, 53(3):468–477, Marzo 2006.

[7] Martínez-Sánchez, Francisco: *Trastornos del habla y la voz en la enfermedad de Parkinson*. Revista de Neurología, 51(9):542–550, 2010.

[8] Tsanas, Athanasios, Max A. Little, Patrick E. McSharry y Lorraine O. Ramig: *Accurate Telemonitoring of Parkinson's Disease Progression by Noninvasive Speech*

Tests. IEEE TRANSACTIONS ON BIOMEDICAL ENGINEERING, 57(4):884–893, 2010.

[9] Del Pino, P.: *Identificación de los Parámetros Espectrales que Determinan la Calidad de la Voz*. Tesis de Maestría, Universidad de Carabobo, 2003.

[10] Jiménez, C., J.A. Díaz, P. Del Pino y H. Rothman: *Aplicación de la Transformada de Wavelet para el Análisis de Señales de Voz Normales y Patológicas*. Revista Ingeniería UC, 15(1):7–13, 2008.

[11] Bernal, J., P. Gómez y J. Bobadilla: *Una Visión Práctica en el Uso de la Transformada de Fourier Como Herramienta Para el Análisis Espectral de la Voz*. Estudios de Fonética Experimental. Universitat de Barcelona, 10:77–105, 1999.

[12] Del Pino, P., I. Granadillo, M. Miranda, C. Jiménez y J.A. Díaz: *Diseño de un Sistema de Medición de Parámetros Característicos y de Calidad de Señales de Voz*. Revista Ingeniería UC, 15(1):13–20, 2008.

[13] Diniz, P.S.R, E.A.B. Da Silva y S.L Netto: *Digital Signal Processing: System Analysis and Design*. Cambridge University Press, 2010.

[14] Madisetti, V: *Digital Signal Processing Fundamentals*. CRC Press, 2010.

[15] Sundararajan, D.: *Digital Signal Processing: Theory and Practice*. World Scientific, 2003.

[16] Boquera, M.C.E.: *Servicios avanzados de telecomunicación*. Díaz de Santos, 2003.

[17] Salcedo, D. y A. Teixeira: *Diseño de un Sistema de Reconocimiento del Habla Para Controlar Dispositivos Eléctricos*. Tekhne. Revista de la Facultad de Ingeniería. UCAB., 10:92–106, 2007.

[18] Cobeta, I., F. Núñez y S. Fernández: *Patología de la voz:*. MARGE BOOKS, 2013.

[19] Jackson-Menaldi, M.C.A.: *La voz normal*. Editorial Médica Panamericana, 1992.

[20] Suárez, C., L.M. Gil-Garcedo, J. Marco, J.E. Medina, P. Ortega y J. Trinidad: *Tratado de Otorrinolaringología y Cirugía de Cabeza y Cuello*. Ed. Médica Panamericana.

[21] Díaz, J.A., C. Sapienza, H. Rothman y Y. Natour: *Algoritmo Robusto Para la Detección de la Frecuencia Fundamental de la Voz Basado en el Espectograma*. Revista Ingeniería UC., 10(3):7–16, 2003.

[22] Raphael, L.J., G.J. Borden y K.S. Harris: *Speech Science Primer: Physiology, Acoustics, and Perception of Speech*. Communication sciences. Lippincott Williams & Wilkins, 2007.

[23] Smith, J.O.I.I.I.: *Spectral Audio Signal Processing*. W3K, 2011.

[24] Gasquet, C. y P. Witomski: *Fourier Analysis and Applications*. Springer-Verlag, 1998.

[25] Roberts, M.J: *Señales y Sistemas*. McGraw-Hill, 2005.

[26] Proakis, John G. y Dimitris G. Manolakis: *Digital Signal Processing*. Prentice-Hall International, Inc, 3ª edición, 1996.

[27] Duhamel, P. y M. Vetterli.: *Fast Fourier Transforms: A Tutorial Review And A State Of The Art*. Signal Processing, 19(4):259–299, Abril 1990.

[28] Deng, L. y D. O'Shaughnessy: *Speech Processing: A Dynamic and Optimization-Oriented Approach*. Taylor & Francis, 2003.

[29] Sánchez, Christian Duque y Mauricio Morales Pérez: *Caracterización de voz empleando análisis tiempo-frecuencia aplicada al reconocimiento de emociones*. Trabajo de grado para optar al título de Ingeniero Electricista. Universidad Tecnológica de Pereira, Abril 2007.

[30] Zamorano, M.: *Análisis de Señales Mediante STFT y Wavelet. Aplicación a Defectología en Rodamientos*. Proyecto fin de carrera para obtener el título de Ingeniero Industrial, 2010.

[31] Francisco, Rodríguez y Tulio Loreto: *Implementación de un detector de frecuencia fundamental de voz en tiempo real usando la plataforma STM32F407 Discovery de STMICROELECTRONICS*. Universidad de Carabobo, 2014.

[32] Fulop, S.A.: *Speech Spectrum Analysis*. Signals and Communication Technology. Springer, 2011.

[33] Batalla, F.N. y C.S. Nieto: *Espectrografía clínica de la voz*. Universidad de Oviedo, 1999.

[34] Peterson, Pearu: *F2PY: a tool for connecting Fortran and Python programs*. Int. J. Computational Science and Engineering, 4(4):296–305, 2009.

[35] Jackson-Menaldi, M.: *La Voz Patológica*. Ed. Médica Panamericana, 2002.

[36] Morrison, Murray y Linda Rammage: *Tratamiento de los trastornos de la voz.* Elsevier España, 1996.

[37] Ramírez, C., E. Sacristán, N. Sáenz, V. Osma y J. I. Godino: *Manual de Usuario MediVoz Captura*. Universidad Politécnica de Madrid, 2003.

[38] Baken, R.J. y R.F. Orlikoff: *Clinical Measurement of Speech and Voice*. Speech Science. Singular Thomson Learning, 2000.

Anexo A

Manual de usuario del software desarrollado

Anexo B

Copia de reporte de resultados